ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΤΜΗΜΑ ΟΙΚΟΝΟΜΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

«ΔΙΟΙΚΗΣΗ, ΑΝΑΛΥΤΙΚΗ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ ΕΠΙΧΕΙΡΗΣΕΩΝ»

Master of Science in

Business Administration, Analytics and Information Systems

Master Thesis

Machine Learning and classification of transport mode choice using Python

Marios Melachroinos

Supervisor

Thanasis Argyriou

Athens 2024

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΤΜΗΜΑ ΟΙΚΟΝΟΜΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

«ΔΙΟΙΚΗΣΗ, ΑΝΑΛΥΤΙΚΗ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ ΕΠΙΧΕΙΡΗΣΕΩΝ»

Master of Science in

Business Administration, Analytics and Information Systems

Master Thesis

Machine Learning and classification of transport mode choice using Python

Marios Melachroinos

Supervisor

Thanasis Argyriou

Athens 2024

The approval of this thesis by the Department of Economics (MSc Administration, Analytics and Information Systems) of the National and Kapodistrian University of Athens does not necessarily imply the acceptance of the author's views on behalf of the Department.

I hereby certify the submitted thesis and the work presented is personal and that all sources and materials used have been properly referenced in the text and bibliography.

Marios Melachroinos

Acknowledgements

I would like to express my gratitude to my family for their continuous assistance throughout this journey. Their belief in my capabilities and constant motivation provided the strength needed to overcome the challenges of completing this master's thesis.

I extend my appreciation to Professor Thanasis Argyriou for his guidance and mentoring during this research journey. His expertise, constructive feedback, and dedication to academic excellence have significantly influenced the quality of this thesis.

My heartfelt thanks also go to my friends and colleagues who generously provided their perspectives and assistance at various stages of this research. Their collaboration added depth to the project and enriched the overall learning experience.

This thesis is dedicated to all those who supported and inspired me. Their contributions have played a pivotal role in the completion of this academic pursuit.

CONTENTS

TABLE OF FIGURES

LIST OF TABLES

ABSTRACT

Urban transportation systems play a pivotal role in shaping modern cities, influencing economic productivity and environmental sustainability. As cities grapple with expanding populations and evolving transportation options, understanding the dynamics behind individuals' mode choices becomes crucial. This thesis navigates the intersection of urban mobility and data science, employing machine learning and Python programming to classify transport mode choices. While traditional methods rely on statistical analyses, machine learning offers a good alternative, by enhancing accuracy through vast dataset analysis.

The research focuses on two distinct cases: Thessaloniki, Greece, and the Netherlands. Thessaloniki serves as a case study on how residents commute to their workplace. Four models – Decision Tree, Random Forest, XGBoost, and a Stacked model – were employed, initially trained with 19 features and later with the most important features. Results indicate that Random Forest and XGBoost exhibit superior performance, with Distance and Time emerging as critical features.

In the Netherlands, the study analyses the trips of residents throughout the country, leveraging a large dataset from the Bureau of Statistics. Undersampling and oversampling techniques address class imbalances in the dataset, which consists of about 800,000 trips and 20 features. Logistic Regression, Decision Tree, Random Forest, and XGBoost models were applied, considering metrics suitable for imbalanced datasets. Results show that XGBoost and Random Forest consistently outperform Logistic Regression and Decision Tree, emphasizing the significance of Distance, Duration, Age, and Car Ownership.

This research contributes valuable insights into predicting transport mode choices, acknowledging the limitations inherent in the specificity of models to contexts. The findings highlight the need for an effective and efficient approach to address diverse urban scenarios, laying the groundwork for future research endeavours in the dynamic field of urban mobility and machine learning applications.

Keywords: Python, Transport Mode Choice, Machine learning, Classification, Classification Evaluation Metrics

ΠΕΡΙΛΗΨΗ

Τα συστήματα αστικών μεταφορών διαδραματίζουν κεντρικό ρόλο στη διαμόρφωση των σύγχρονων πόλεων, επηρεάζοντας την οικονομική παραγωγικότητα και την περιβαλλοντική βιωσιμότητα. Καθώς οι πόλεις παλεύουν με τους αυξανόμενους πληθυσμούς και τις εξελισσόμενες επιλογές μεταφοράς, η κατανόηση της δυναμικής πίσω από τις επιλογές τρόπων μετακίνησης των ατόμων γίνεται κρίσιμη. Αυτή η διατριβή επικεντρώνεται στην αστική κινητικότητα και της επιστήμης δεδομένων, χρησιμοποιώντας μηχανική μάθηση και προγραμματισμό Python για την ταξινόμηση των επιλογών τρόπου μεταφοράς. Ενώ οι παραδοσιακές μέθοδοι βασίζονται σε στατιστικές αναλύσεις, η μηχανική μάθηση προσφέρει μια καλή εναλλακτική λύση, ενισχύοντας την ακρίβεια μέσω ευρείας ανάλυσης δεδομένων. Η έρευνα επικεντρώνεται σε δύο διακριτές περιπτώσεις: τη Θεσσαλονίκη και την Ολλανδία. Η Θεσσαλονίκη χρησιμεύει ως μελέτη περίπτωσης για το πώς οι κάτοικοι μετακινούνται στον χώρο εργασίας τους. Χρησιμοποιήθηκαν τέσσερα μοντέλα – Decision Tree, Random Forest, XGBoost και Stacked model, τα οποία αρχικά εκπαιδεύτηκαν με 19 χαρακτηριστικά και αργότερα με τα πιο σημαντικά χαρακτηριστικά. Τα αποτελέσματα υποδεικνύουν ότι το Random Forest και το XGBoost παρουσιάζουν ανώτερη απόδοση, με την απόσταση και τον χρόνο να αναδεικνύονται ως κρίσιμα χαρακτηριστικά. Στην Ολλανδία, η μελέτη αναλύει τα ταξίδια των κατοίκων σε όλη τη χώρα, αξιοποιώντας ένα μεγάλο σύνολο δεδομένων από την Στατιστική υπηρεσία της Ολλανδίας. Οι τεχνικές υποδειγματοληψίας και υπερδειγματοληψίας αντιμετωπίζουν τις ανισορροπίες κλάσεων στο σύνολο δεδομένων, το οποίο αποτελείται από περίπου 800.000 διαδρομές και 20 χαρακτηριστικά. Εφαρμόστηκαν μοντέλα Logistic Regression, Decision Tree, Random Forest και XGBoost, λαμβάνοντας υπόψη μετρήσεις κατάλληλες για μη ισορροπημένα σύνολα δεδομένων. Τα αποτελέσματα δείχνουν ότι το XGBoost και το Random Forest ξεπερνούν σταθερά τα μοντέλα Logistic Regression και το Decision Tree, δίνοντας έμφαση στη σημασία της απόστασης, της διάρκειας, την ηλικία και της ιδιοκτησίας αυτοκινήτου. Αυτή η έρευνα συνεισφέρει πολύτιμες γνώσεις για την πρόβλεψη των επιλογών του τρόπου μεταφοράς, αναγνωρίζοντας τους περιορισμούς που είναι εγγενείς στην ιδιαιτερότητα των μοντέλων στα πλαίσια. Τα ευρήματα υπογραμμίζουν την ανάγκη για μια αποτελεσματική και αποδοτική προσέγγιση για την αντιμετώπιση διαφορετικών αστικών σεναρίων, θέτοντας τις βάσεις για μελλοντικές ερευνητικές προσπάθειες στο δυναμικό πεδίο της αστικής κινητικότητας και των εφαρμογών μηχανικής μάθησης.

Λέξεις-κλειδιά: Python, Επιλογή τρόπου μεταφοράς, Μηχανική μάθηση, Ταξινόμηση, Μετρήσεις αξιολόγησης ταξινόμησης

1.INTRODUCTION

Urban transportation systems serve as the foundation of modern cities, exerting influence on aspects ranging from economic productivity to environmental sustainability. In the face of global challenges related to expanding urban populations and the continuous evolution of transportation options, identifying the dynamics underlying individuals' choices in transportation modes has become increasingly vital. This thesis explores urban mobility and data science, employing machine learning techniques and the programming language Python to categorize transport mode choices. In a dynamic urban landscape where transportation modes have expanded beyond traditional choices like cars and public transit to include various options such as ridesharing and mobility solutions, traditional methods of understanding and predicting transportation behaviours through statistical analyses have been challenged. The emergence of machine learning provides an alternative, offering the ability to analyse extensive datasets and diverse patterns, presenting a new opportunity to increase the accuracy and efficiency of predicting transport mode choices. Despite the increased accuracy achieved by machine learning models in predicting transport mode choices, their applicability is often constrained to specific contexts. A model designed for a particular transport behaviour may struggle to generalize well to other scenarios. This limitation originates from significant variations in infrastructure, city layouts, and resident preferences across different regions. To attain a comprehensive understanding of transport mode behaviour and patterns, it is imperative to apply models to diverse cases and scenarios.

This Thesis delves into the realm of machine learning and classification of transport mode choice with a specific focus on two distinct cases. The first case concentrates on how residents in the city of Thessaloniki commute to their workplace. Despite being the second-largest city in Greece and hosting major transportation hubs like a significant port and an airport, Thessaloniki's infrastructure is often characterized by mediocrity, offering only buses as the available option for public transportation. The second case concentrates on the commuting behaviour in Netherlands, analysing the trips conducted by the residents throughout the country. The primary goal of this thesis is to utilize machine learning techniques, leveraging Python programming language, to predict commuting behaviour in Thessaloniki and Netherlands. Python is widely acknowledged for its simplicity and ease of comprehension for the reader. The main strength of the language is the extensive pool of libraries and packages that constitute machine learning techniques and other developing tasks easier to implement (python.org, 2024).

The subsequent chapters will unfold a comprehensive exploration of existing literature, laying the groundwork for understanding the historical context and evolution of transportation mode choice research. Additionally, the terminology, models, and evaluation metrics in machine learning will be explored upon, drawing insights from literature papers and books. The methodology section will detail the approach taken in data collection process for both cases, model selection and evaluation techniques,

followed by a comprehensive exploratory data analysis. Subsequent chapters will concentrate on the practical application of models and the interpretation of their outputs. Ultimately, conclusive remarks will be drawn regarding the results and limitations of the study, accompanied by recommendations for future research and endeavours.

2. LITERATURE REVIEW

Forecasting an individual's choice of transportation mode has garnered significant academic interest in recent years. The field of transportation and behavioural analysis has primarily relied on the extensive utilisation of logit models in existing research. In general, logit models and logistic regression analysis have been employed, specifically to examine the connection between the likelihood of binary or ordinal responses and explanatory variables using the maximum likelihood estimation method (Trueck, 2009). According to Trueck (2009), the logistic function is given by the following expression:

$$P(Y = 1 \mid x_1, \ldots, x_k) = \frac{exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n)}{1 + exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n)}.$$

The above can also be rewritten as:

$$logit(P(Y = 1 \mid x_1, \ldots, x_k)) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

Where βo, β1, β2…….βn represent the parameters of the corresponding independent variables x1, x2……xn in the dataset.

McFadden (1972, 1974) posed a significant challenge to the initial approach to comprehending travel behaviour. In his exploration of the analysis of transit behaviour, he advocated for the application of the multinomial logit model. Like binary logit models, the multinomial model adheres to the same principles and assumptions, with the notable difference being the presence of multiple alternative choices (Lee & Kim, 2023). Furthermore, because of the assumption of independence among the options, the total probabilities of all choices add up to 1 (Lee & Kim, 2023). Following the introduction of the logit model, subsequent research endeavours have sought to extend this model and investigate travel behaviour, thereby addressing certain constraints associated with the original logit model. Such examples are the “nested logit model” (Willis, 2014) and the “mixed logit model” proposed by McFadden & Train (2000). The nested logit model organises choices into separate nests and permits varying correlations between these nests. As a result, correlations are consistent within each nest, but for options situated in different nests, the unobservable elements are uncorrelated and, in fact, entirely independent (Willis, 2014). The mixed logit model is defined as a standard multinomial model, also called “latent class model”, where the coefficients are chosen from a cumulative distribution, introducing an element of randomness (McFadden & Train, 2000).

2.1. Factors influencing transport mode selection.

Since the appearance of logit models, numerous studies have surfaced with the objective of delving into the crucial factors that affect transportation mode selection, going beyond the conventional determinants like cost or travel distance. In a study conducted by Mayo and Taboada in 2020, they

employed a hierarchy model to assess the determinants affecting the selection of public transportation mode among respondents in the Philippines. Their survey results indicated that safety was the most significant consideration, followed by cost, comfort, and concerns about environmental sustainability (Mayo & Taboada, 2020). Another study by Donkor et al. in 2020 examined the role of emotions in transport mode selection, focusing on respondents in the city of Edinburgh. Their findings revealed that an individual's feelings and experiences related to public transportation, along with their socio-demographic characteristics, exerted a substantial influence on their transit behaviour (Donkor et al., 2020). Additionally, McCarthy et al. in 2017 conducted further research suggesting that the presence of young children in a family had an impact on transportation behaviour. Specifically, they proposed that families with children preferred car usage over other sustainable transit options, with psychosocial factors and household characteristics playing pivotal roles in the choice of transportation mode (McCarthy et al., 2017). The influence of various weather conditions in transport mode choice have also been explored in the literature. Bocker et al. (2016) delved into weather-related factors and their connection to transit choices. Based on their analysis of travel diaries in the Netherlands, their results indicate that individuals who opt for walking or cycling modes are particularly affected by weather conditions (Bocker et al., 2016).

Additional research in the field of transportation focuses on mode selection during the COVID-19 pandemic and its impact on individual transit choices. To elaborate, Mussone & Changizi (2023) conducted a study that utilised a logistic regression model to investigate the factors influencing transportation mode choices prior, during, and post COVID-19 lockdowns. Their research, based on data from residents in Milan, Italy, indicated that socio-demographic factors, as well as individual preferences and concerns related to public transportation, played the most significant roles in predicting transport mode choices during the pandemic (Mussone & Changizi, 2023). It is worth noting that, despite mandatory contamination control measures, many residents in various countries expressed heightened concerns about the spread of the virus within public transportation during and after lockdown restrictions. Those concerns led residents to transition from relying on public transportation to utilising private vehicles for their transit requirements. The phenomenon is further investigated by Das et al. (2021), who employed a logistic regression model to examine travel behaviour and modal transitions. Their research findings indicate that demographic factors exert a considerable influence on preferences for switching transportation modes. Additionally, trip-related factors, including travel time and health conditions, demonstrate a robust association with the inclination to substitute public transportation for car usage (Das et al., 2021).

2.2. Machine Learning essentials

An alternative to the traditional logit models, comes with the introduction of machine learning. According to Zhou (2021), Machine learning is a method that enhances the performance of systems

through computational learning from prior experiences. In the realm of computer systems, these experiences are the data themselves. The central objective of machine learning is to create algorithms capable of constructing models based on this data. When the algorithm is supplied with experiential data, it yields a model capable of making predictions for new observations (Zhou, 2021). Based on the presence or absence of labelled training data, problems can be categorised into two groups: supervised and unsupervised learning. Supervised learning encompasses a training phase in which the algorithm is supplied with a dataset comprising pairs of input and corresponding output (referred to as labelled data). During this phase, the algorithm acquires the ability to make predictions or classifications by drawing insights from this labelled data (Zhou, 2021). In supervised learning, the primary tasks are categorised into regression and classification, depending on whether the prediction output is continuous or discrete. In the case of classification problems, when there are only two possible labels, it is referred to as a "binary classification problem" (Zhou, 2021). If there are multiple possible labels, it is termed a "multi classification problem" (Zhou, 2021). Common algorithms that aim to solve regression and classification problems include NBC, KNN, Decision Trees, Ensemble Learning, Boosting, Support Vector Machines and Neural Networks (Kubat, 2021). The plot below illustrates the difference between regression and classification.

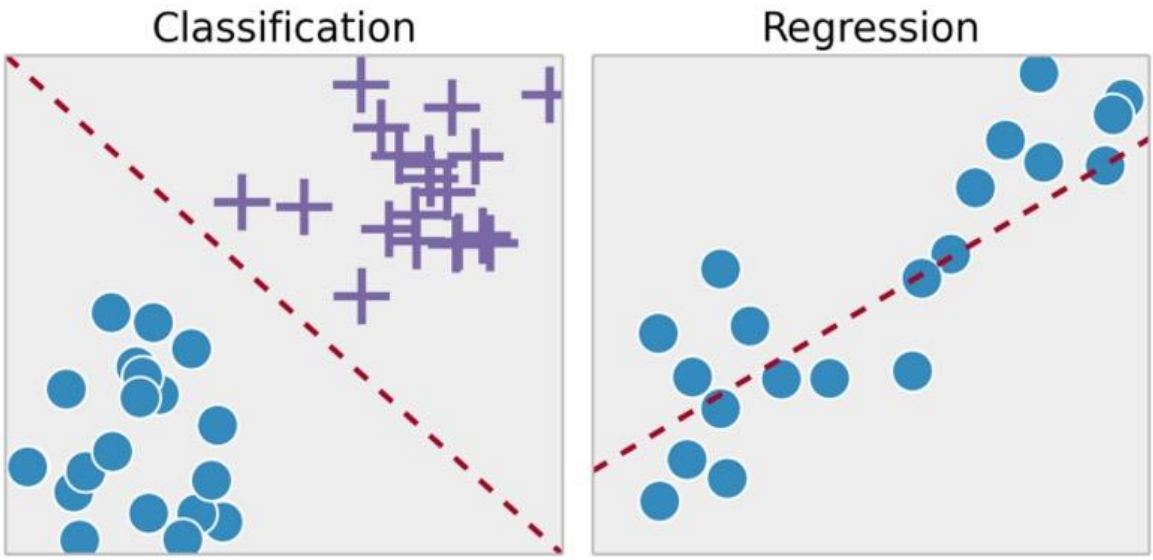


Figure 1. Classification vs Regression

Source: Svitla.com, 2023

.

In contrast, unsupervised learning is a category of machine learning in which the algorithm is given input data but does not have access to predefined output labels. In this context, the algorithm's role is to autonomously identify patterns, structures, or relationships within the data, all without prior knowledge of the expected output (Zhou, 2021). The main task in unsupervised learning is clustering

which involves the process of categorising data points by identifying their similarities (Zhou, 2021). The most common technique for such problems is K-Means Clustering.

There is also a third form of learning called “Reinforcement learning” (Kubat, 2021). In this domain, the objective differs significantly. Here, the agent's role is not to induce knowledge from a pre-classified dataset but to engage in active experimentation with a system. The system, in turn, provides feedback in the form of rewards or penalties in response to the agent's actions. The agent's primary aim is to refine its behaviour by seeking to maximise rewards and minimise penalties as it interacts with the system (Kubat, 2021).

2.3. Classification Algorithms & Techniques

The most common algorithms and machine learning techniques that aim to solve classification tasks in a supervised problem are presented as followed:

K-Nearest Neighbour (KNN)

Perhaps the easiest and most straightforward algorithm for classification is that of KNN. The algorithm involves determining the class of a sample by identifying its closest neighbours and utilizing their characteristics. The term "k-Nearest Neighbour" is used because, in many cases, it is essential to consider more than just a single neighbour (k) to classify an example (Cunningham & Jane, 2021). Identifying the nearest neighbour is accomplished through the application of "Euclidean Distance" (Akanbi, 2014). The process in the algorithm, as well as the equation employed, are outlined as follows:

$$dx_i, x_j = \sqrt{\sum_{r-1}^{n} a_r x_i - a_r x_j{}^2}$$

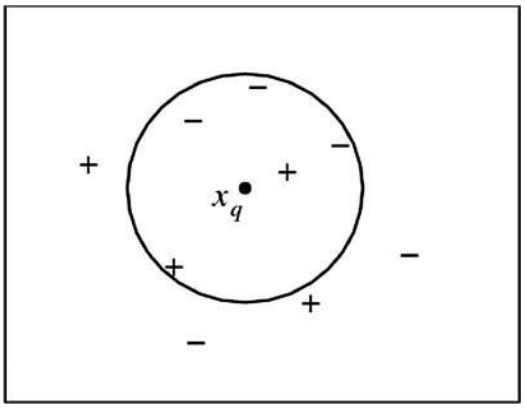


Figure 2. KNN

Source: Akanbi & Fazeldehkordi, 2014

Naïve Bayes Classifier (NBC)

NBC is a machine learning model that implements the Bayes probability theory. Through the examination of the input data corresponding to a specified set of features, the Naïve Bayes classifier

can calculate class probabilities associated with a specific label (Souza et al, 2021). To classify the input data, it is necessary to calculate the probabilities associated with each existing label. The label assigned with the highest probability is then identified as the one to which the input data belongs (Souza et al, 2021). Thus, the classification task is to locate the class with the maximum probability to occur. The Bayes theory along with the max probability equation are presented below:

$$PAB = \frac{PBAPA}{PB} \qquad a = \text{argmax}_a Pab_1 \cdots b_n$$

Source: Souza et al, 2021

Decision Trees (DT)

Decision trees serve as a versatile approach usable to both regression and classification problems. They stand out as one of the most employed algorithms, particularly suitable for classification tasks, offering numerous advantages over alternative classifiers. They are relatively easy to explain and interpret while requiring little effort for data preparation from the user (Benferhat & Elouedi, 2006). The figure below illustrates the basic structure of a decision tree.

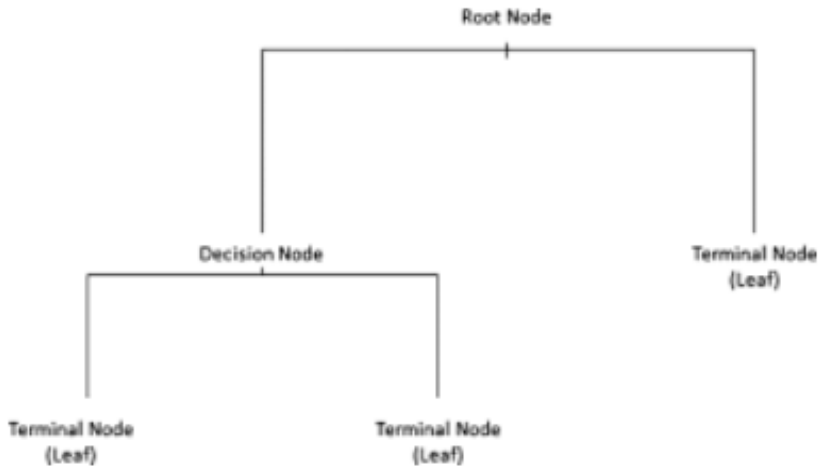


Figure 3. Decision Tree

Source: Bellini, 2019

On a tree structure the root represents the starting point of the decision-making process. The branches represent the splits/path of the root node, while decision nodes evaluate the features to split the data and which direction to follow. Lastly, the terminal nodes represent the outcome, or while in classification task, the class label (Bellini, 2019). Two terms used for decision trees are Gini impurity and entropy.

Gini measure the probability that a random instance is misclassified when chosen randomly. Lower values of Gini indicate a lower likelihood for misclassification (Dash, 2022). The formula of Gini is:

$$Gini = 1 - \sum_{i=1}^{j} P(i)^2$$

Likewise, entropy is a measure of 'disorder' in a node with values ranging from 0 to 1 (Dash, 2022). The formula of entropy is:

$$E = -\sum_{i=1}^{n} p_i \, log_2(p_i)$$

A useful technique that is associated with decision trees is that of pruning, which is implemented in situations that the decision tree may grow too long, so it is essential to reduce its decision nodes and avoid overfitting on the data (Bellini, 2019).

Random Forest

Frequently, it is crucial to construct more complex models by combining the predictions of several weaker models. This approach in data science is commonly referred to as the "ensemble" method. This procedure essentially combines different models that are trained to address the same issue and achieve improved outcomes, making the final model more robust (Rocca, 2019).

Random Forest stands out as the most widely adopted ensemble method for classification tasks, a model that trains multiple decision trees through a process called "bootstrapping", followed by another process known as "bagging" aggregation. Bootstrapping refers to the training of individual trees simultaneously and on distinct subsets of the training set, utilising a diverse set of features. This process ensures that each decision tree in the random forest differs, which consequently reduces the variance of the model. For the model to make the final decision, the predictions from each individual tree are aggregated leading to a more generalised method. Due to its robust nature, the random forest often surpasses other classifiers in terms of accuracy. Despite being more complex in structure compared to an individual decision tree, the random forest is generally simpler when it comes to hyperparameter tuning (Misra & Li, 2020). A typical structure of the model can be viewed in the following image.

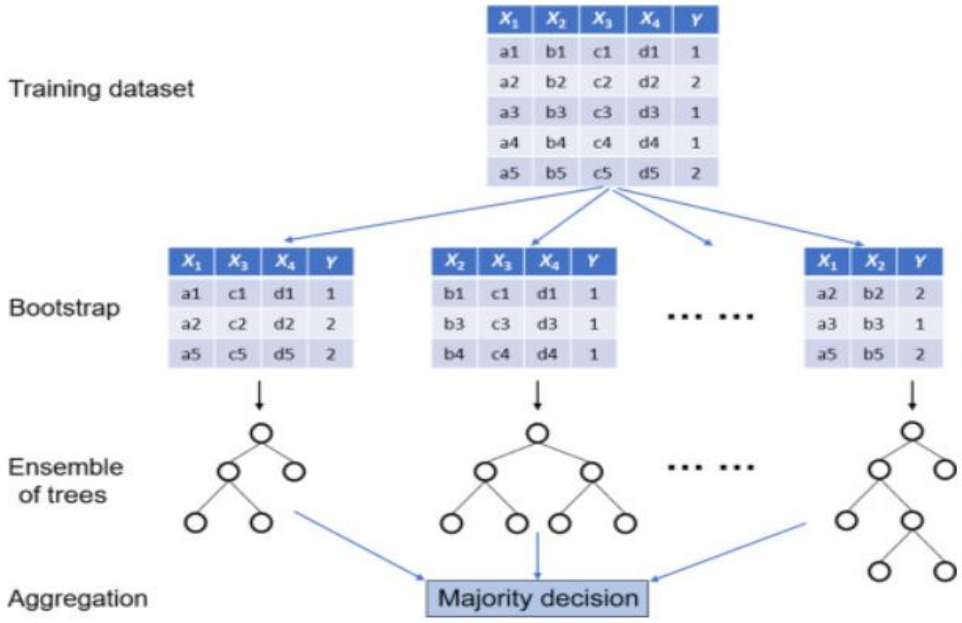


Figure 4. Random Forest

Source: Misra & Li, 2020

Boosting

Another well-known ensemble technique that combines numerous "weak learners" to create an enhanced model with improved accuracy and robustness. Comparable to the bagging technique of the random forest, boosting also involves aggregating predictions from each individual decision tree, resulting in a more generalised model. However, the primary distinction lies in the fact that, during boosting, the trees are trained sequentially, unlike the parallel training that takes place in the random forest. In this process, each model in the sequence is fitted with a focus on observations in the dataset that were mishandled by the preceding learners. Essentially, each new learner concentrates its efforts on the most challenging instances encountered so far. Consequently, by the end of the process, a robust classifier with reduced bias is obtained (Rocca, 2019). The process of boosting compared to bagging can be observed in the figure below.

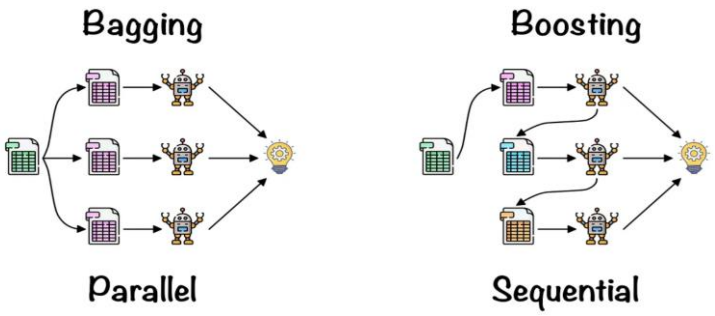


Figure 5. Boosting vs Bagging

Source: Lopez, 2021

Contrary to bagging and random forest, boosting exhibits numerous variations in the form of various algorithms. Such examples are:

- ***Adaptive Boosting***: The concept of adaptive boosting is to train multiple weak classifiers to create a robust classifier. In each iteration, a weak learner is trained on a weighted version of the train set. The weight assigned to each iteration is adjusted based on the preceding learner. Adaptive boosting is implemented by the Adaboost Classifier algorithm (Polamuri, 2023).
- ***Gradient Boosting***: The process starts by training a weak learner using the initial data. It subsequently calculates the residuals, and another weak learner is fitted to the residuals. This process is repeated for a set number of iterations, where each new model aims to minimise, the residuals left by the preceding learner. The prediction of the model is the cumulative sum of predictions made by the weak learners (Polamuri, 2023).
- ***XGB (Extreme Gradient Boosting)***: An enhanced version of gradient boosting, this model has emerged as one of the most widely adopted methods for boosting procedures. This algorithm facilitates considerably faster training by enabling parallel processing and is adept at handling very large datasets with minimal requirements for data preprocessing. Like other boosting principles, XGBoost trains multiple week decision trees to create a strong model with increased accuracy and robustness (GeeksforGeeks, 2023).

Stacking

A third ensemble method is that of stacked generalization. In this approach multiple models are trained on the same dataset serving as base estimators or "level-0 base models". The predictions of those models are then used to train another model called as "level-1 meta estimator". Hence, the output of the base models (predictions) is used as an input to the meta estimator. The main idea behind stacking generalization is to combine diverse model and combine the strengths of all those models. The meta estimator is then trained to combine those strengths for an enhanced predictive power (Brownlee, 2021). The picture below illustrates the procedure of how stacking occurs. First the training set is split into folds. K-1 folds are used to train the base models, while the validation fold is used for predictions. The predictions from all the base models are then stacked and feed the meta learner who is responsible to make the final prediction.

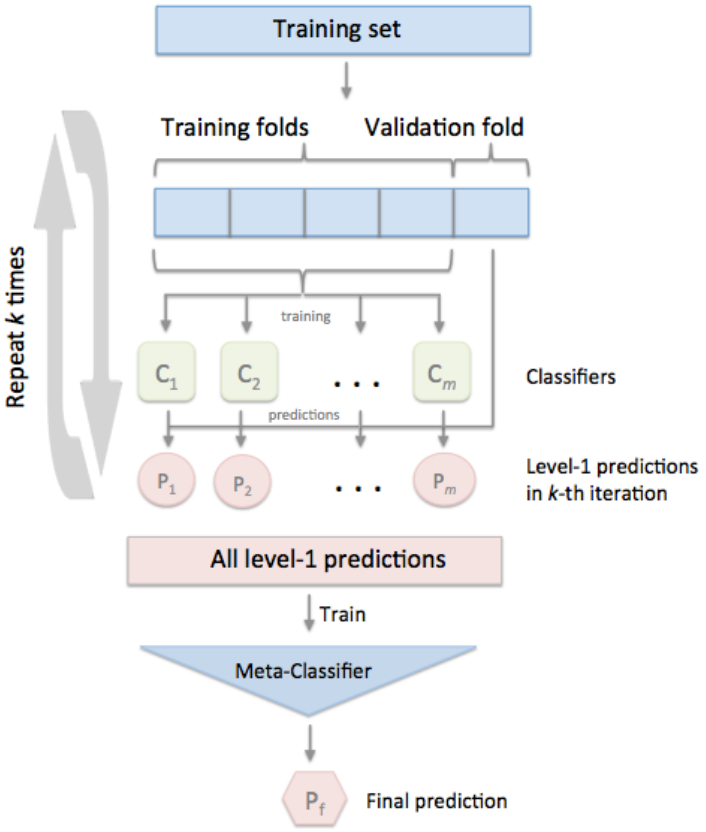


Figure 6. Stacking

Source: StackingCVClassifier: Stacking with cross-validation - mlxtend (rasbt.github.io)

2.4. Class imbalance and resampling strategies

On many occasions, real world data comes with an issue called class imbalance. This occurs primarily on classification tasks when the distribution of the target variable is severely skewed, meaning that a certain class is underrepresented than others. Such situations occur often in disease detection and fraud transaction datasets where there are significantly fewer observations for diseased or fraud cases. This skewness often creates challenges for the models since they become more biassed towards the majority class, resulting in poor performance for the minority classes (Yang et al, 2024). To address this situation, resampling techniques are employed during the model training process, aiming to balance the distribution and minimise any bias in the model. The most widely used techniques are random **Undersampling** and **Oversampling**. In random undersampling, samples are deleted from the majority class until it matches the minority, though it can result in loss of information and underfitting. In contrast, with random oversampling, samples from the minority class are duplicated until it matches with the majority class. However, this considerably inflates the dataset size and increases the risk of overfitting in the training data. To mitigate such scenarios, methods like cross-validation, hyperparameter tuning, and regularisation are employed to minimise issues related to underfitting and overfitting. It is crucial though, that both of those techniques are only applied on the train set, since the

test set serves as a genuine representation of the real-world problem and should always be left unaltered (Brownlee, 2021).

2.5. Evaluation metrics

The ultimate objective of any classification model is to generate predictions for new and unseen data. Hence, to assess whether a built model is effective, various evaluation methods are necessary to make a concrete decision. Those metrics are presented as followed:

Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP = True positive; FP = False positive; TN = True negative; FN = False negative

Typically, accuracy is calculated as the ratio of correct predictions made to the total number of instances. However, relying solely on accuracy as a metric can be overly optimistic, especially in situations with significant class imbalance. For instance, in a scenario with 1000 transactions, of which only 100 are fraud, a classifier could predict all cases as non-fraudulent, resulting in a 90% accuracy without detecting any of the fraud cases. To address this, additional metrics are considered that can emphasising on the evaluation of minority classes.

Recall

Also referred to as sensitivity, recall measures how many of the results, that a model returned, were relevant. It is calculated as the ratio of true positives to the number of true positives plus the number of false negatives.

$$\text{Recall} = \frac{tp}{tp + fn}$$

Source: Wikipedia, 2024

Recall is very used when you must correctly classify some event that has already occurred. It is frequently applied in scenarios such as fraud detection models or disease detection in patients. For instance, in the context of illness detection, it is crucial to identify individuals who are ill to prevent false negatives.

Precision

Precision is a measure of result relevancy. It is calculated as the ratio of true positives over to the number of true positives plus the number of false positives.

$$\text{Precision} = \frac{tp}{tp + fp}$$

Source: Wikipedia, 2024

Precision is widely employed in marketing campaigns, particularly in the context of marketing automation. This is because a marketing automation campaign aims to initiate an activity for a user when the system predicts a successful response. In these scenarios, low precision translates to a financial loss as it entails reaching out to potential customers who are not interested in the marketing offer.

F1 score

F1 calculates the trade-off between precision and recall. It is often preferable in situations where the dataset is imbalanced, while both high recall and precision are desirable. The F1 score is expressed by the following equation.

$$F_1 = \frac{2T_p}{2T_p + F_p + F_n}$$

It can also be expressed in terms of precision and recall by the following expression.

$$2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Source: Wikipedia, 2024

Receiver Operating Characteristic (ROC) Curve

The ROC curve illustrates the balance between the true positive rate (Sensitivity or Recall) and the false positive rate (FPR) across various probability thresholds. AUC (Area Under the Curve) is a metric assessing the models capability to discriminate between those two. Its value ranges from 0 to 1, with values nearing 1 signifying optimal classifier performance. While commonly employed for binary classification tasks, it is possible to generate the curve for multiclass problems as well. In this context, each class has its individual curve, while Macro-AUC represents the average of the class-specific AUC values. An example of suck curve is depicted below.

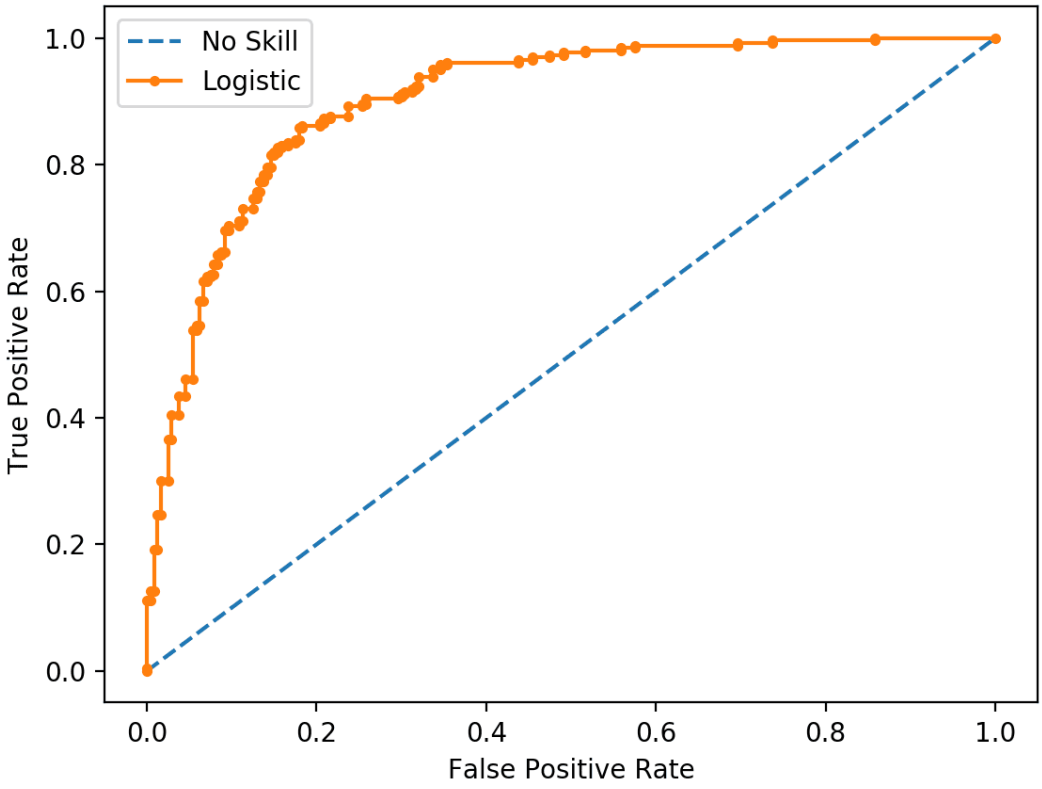


Figure 7. ROC Curve

Source: Brownlee, 2020

Average Precision (AP)

The relationship of recall and precision is depicted by the precision recall curve, which illustrates the trade-off between the two metrics for different probability thresholds. The area under the curve is known as Average Precision (AP) and is given by the following expression (Kashifi et al, 2022).

$$\mathrm{AveP} = \int_0^1 p(r)dr$$

The area under the curve, like the ROC curve, has a value ranging from 0 to 1, with higher values indicating superior classifier performance. Precision-recall curves are preferred in scenarios with severe class imbalance since AUC in those occasions can be overly optimistic. Although commonly employed for binary classification, it is feasible to calculate curves for each individual class in a one-vs-rest (OvR) scenario. The micro-AP is then computed which is the harmonic mean of the class-specific AP values.

The pr curves are preferred in situations where there is a class imbalanced as they are more indicating for the performance of the minority classes. A typical pr curve is illustrated below.

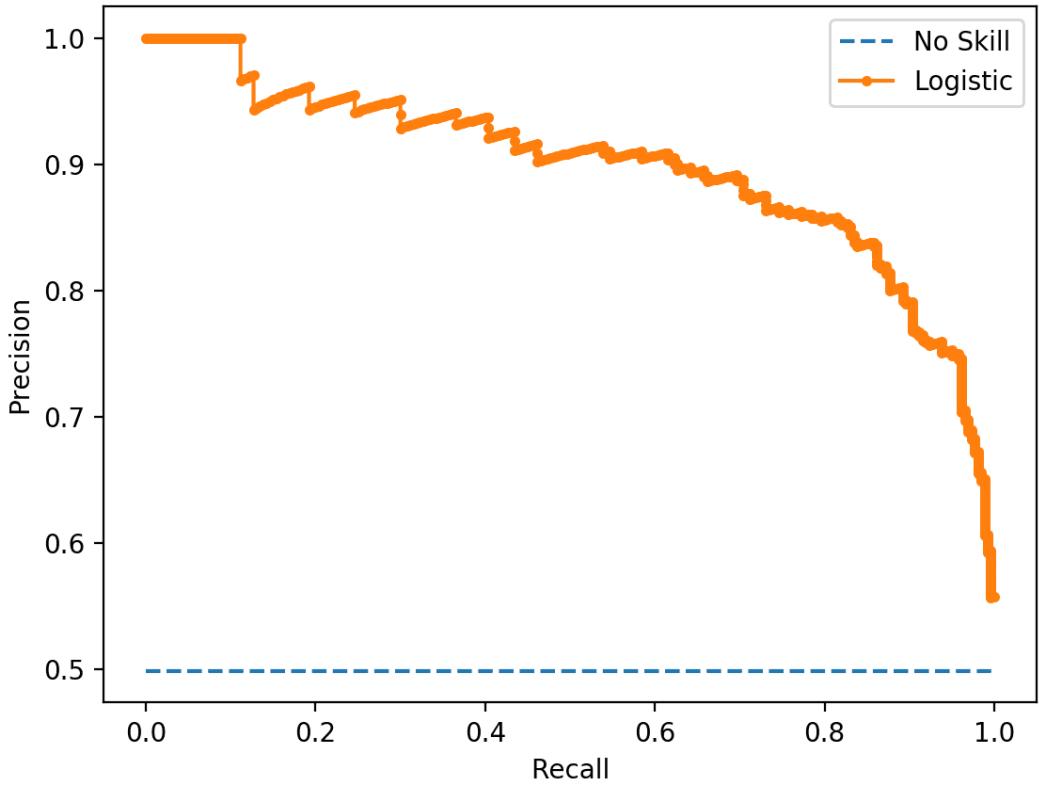


Figure 8. Precision - Recall Curve

Source: Brownlee, 2020

2.6. Machine Learning applications in transport mode choice

The adoption of machine learning in the modelling and prediction of transport mode choices has experienced a significant surge in recent times. In contrast to the prior logit models, the results indicate a notably improved ability to accurately predict transportation mode. This application is geared towards improving our understanding and predictive capabilities concerning individuals' decisions regarding the modes of transportation they choose. Numerous recent studies have sought to employ a range of machine learning models to predict transit behaviour. In 2015, Omrani conducted a study with the objective of forecasting the travel mode choices of individuals by applying machine learning techniques to national data from Luxembourg. His research outcomes revealed that artificial neural networks outperformed other alternative models in terms of predictive accuracy (Omrani, 2015). In their research conducted between 2010 and 2012 using data from the National Travel Survey in the Netherlands,

Hagenauer and Helbich (2017) applied various machine learning classification models to predict travel mode choices effectively. Their results indicated that the Random Forest model outperformed the others. Nonetheless, they noted that while trip distance emerged as the most critical predictor, the importance of variables varied among different classifiers and class labels (Hagenauer & Helbich, 2017). In a parallel study using the same data, Kashifi et al (2022) set out to also forecast transportation mode choices. They extended the previous research by incorporating additional machine learning techniques into their analysis. Their results revealed that boosting and LightGBDT exhibited superior predictive accuracy for the different classes, especially when utilizing both under and oversampling methods to address class imbalance. Furthermore, their analysis underscored that age, income, and distance were the most influential predictors in the context of transport mode prediction (Kashifi et al., 2022).

Other instances focusing on the prediction of transportation modes can be characterised as more tailored to specific scenarios, as they are designed for modelling particular situations. In a recent study, Bhuiya et al (2022) focused on modelling transport mode choices for individuals with limited mobility in Dhaka. Their research findings indicated that multinomial logistic regression and linear discriminant analysis models exhibited superior predictive accuracy, particularly considering a smaller dataset (Bhuiya et al., 2022). Additionally, Zhao et al (2020) investigate differences between machine learning and logit models based on trip diary recordings. Their study findings from staff and students within the University of Michigan suggest that when deciding the two for transport predictions, it seems there is a trade-off between prediction accuracy and the alignment with behavioural principles (Zhao et al, 2020). Recent research developments in this field have introduced more sophisticated approaches, such as the adoption of artificial neural networks and deep learning methods. For instance, Zhang et al. (2020) introduced a deep neural network model for classification using data from Beijing, and their results demonstrate that this network model outperforms the random forest model in predicting transportation modes (Zhang et al., 2020). Furthermore, in a study based on national travel data from the UK, Bei et al. (2023) introduce a deep neural network model that goes beyond mere travel mode prediction, also addressing the purpose of the trip. Their research indicates that the model they proposed surpasses the performance of basic multinomial logit models and single-task neural networks (Bei et al., 2023). Additionally, Wang, Mo, and Zhao (2021) present a "theory-based residual neural network" model that integrates discrete choice models with basic neural networks, using three separate survey datasets. Their results indicate that the model they propose not only achieves superior predictive accuracy but also exhibits greater resilience compared to straightforward neural networks or discrete models (Wang, Mo & Zhao, 2021).

3. Case 1 – Thessaloniki

3.1. Methodology

3.1.1 Python Libraries

Below, the Python libraries utilised for the analysis are displayed:

a) **Numpy**: It is one of the most implemented libraries, particularly in machine learning. This library is known for its support of matrices and multi-dimensional data. Numpy incorporates mathematical functions, among which the Array Interface that stands out as one of its most valuable features.

b) **Pandas**: It stands as a crucial library for data scientists, serving as an open-source machine learning library offering top notch analytical tools. It includes features such as sorting, visualisations, conversions and more.

c) **Matplotlib**: It is employed for visualising numerical data, making it a valuable tool in data analysis. It includes many unique charts such as pie charts, histograms, scatter plots and more.

d) **Scikit-learn**: A library containing most of the classification and regression algorithms. The library works for many separate problems including regression, classification, clustering and more.

e) **Geopy**: A python library that enables users' identification of coordinates for addresses, cities, countries internationally. It is also used for calculating distances between different coordinates (pypi.org, 2024)

f) **Seaborn**: It is used for data visualisation, offering a high-level interface to create visually appealing and informative statistical graphics. The library is built on top of the library of Matplotlib also functioning effectively with Pandas.

g) **Imbalanced-learn**: It is a library that offers various re-sampling techniques specifically designed for datasets with pronounced class imbalance. It is compatible with scikit-learn.

3.1.2 Data collection

Data for Thessaloniki were collected through a survey using the Google Docs platform. While the survey was initially created in Greek, the analysis was conducted in English. The survey was distributed both online through social media and in paper form to residents in Thessaloniki. It was also distributed through the network of History department of the Aristotle University of Thessaloniki and the help of the professors. Responses were also gathered through phone communication and interviewing, where answers were recorded and manually inserted into Forms. The following comprises the complete list of questions posed both in Greek and English

1) Διεύθυνση Κατοικίας - Home Address

2) Διεύθυνση Εργασίας - Work Address
3) Φύλλο - Gender
4) Ηλικία - Age
5) Έχετε δίπλωμα για οποιαδήποτε από τα παρακάτω: Αυτοκινητο, Μηχανη, Φορτηγό, Τίποτα - Do you have a license for any of the following: Car, Motorcycle, Truck, Nothing
6) Διαθέτετε κάποιο από τα παρακάτω: Αμάξι, Μηχανή, Ποδήλατο, πατίνι, τίποτα - Do you have any of the following: Car, Motor, Bicycle, skates, nothing
7) Από πόσα άτομα αποτελείται η οικογένειά σας - How many people does your family consist of ?
8) Πόσα ιδιωτικά οχήματα διαθέτετε στην οικογένειά σας - How many private vehicles do you have in your family?
9) Μηνιαίο εισόδημα - Monthly Income
10) **Με ποιον τρόπο πηγαίνετε συνήθως στην εργασία σας - Mode for commuting to work (target variable)**
11) Αναφέρετε πόσα λεπτά στο περίπου χρειάζεται για να μεταβείτε από το σπίτι σας προς τον χώρο εργασίας σας - Indicate how many minutes approximately it takes you to travel from your home to your workplace.
12) Ποιές ώρες πηγαίνετε συνήθως στην εργασία σας - What time do you usually go to work?

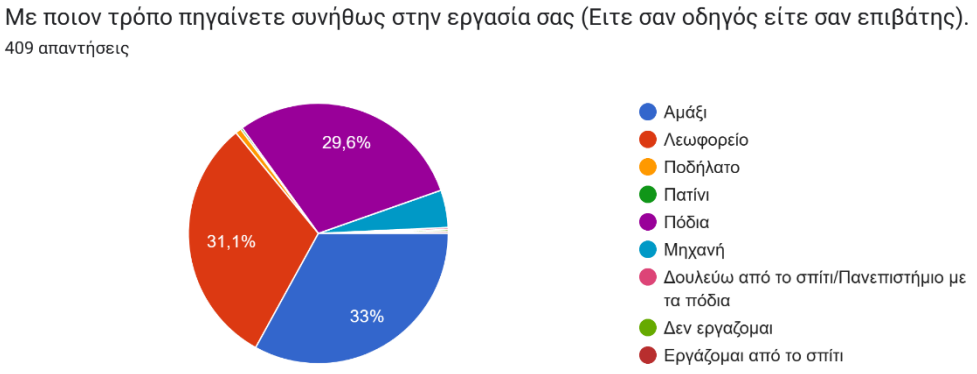
Likert Based questions

Do you think the following factors affect your commute?

Use scale where:

1-Strongly disagree , 2-Disagree , 3-Neutral , 4-Agree , 5-Strongly agree

13) Άνεση Μετακίνησης - Convenience
14) Κόστος Μετακίνησης - Cost
15) Ασφάλεια μετακίνησης - Safety
16) Προστασία του περιβάλλοντος - Environmental concerns
17) Σωματική άσκηση και υγεία - Physical exercise and health
18) Καιρικές συνθήκες - Weather conditions
19) Διαθεσιμότητα χώρου στάθμευσης - Parking availability



The survey remained accessible from November to the end of January and a total of 409 samples were gathered. Looking at the commute mode graph above, the results indicate that most common commute options include bus, motor, car and walk. Other answers involve bike, skate, working from home or not working. Those answers with low frequencies were removed. Additionally, motorcycle and cars were grouped into one category: private vehicle (car/motor). Hence, the analysis procedure was executed considering three possible commute modes: Private vehicle, bus and walk.

After eliminating blank or erroneous responses as well as samples with unclear home or work addresses, 383 were deemed valid for subsequent analysis. The sample was visualized in a Map of Thessaloniki using the home addresses and the use of geopy library in python. The library calculates the coordinates of the home address and creates a map with markers based on those coords. The code used along with the sample visualization are illustrated below.

```
addresses = df['Home_address'].tolist()

# Geocode addresses to obtain latitude and longitude
geolocator = Nominatim(user_agent="address_visualization")
locations = []

for address in addresses:
    location = geolocator.geocode(address)
    if location is not None:
        locations.append(location)

# Create a DataFrame with latitude and longitude
coordinates = [(location.latitude, location.longitude) for location in locations]
df_coordinates = pd.DataFrame(coordinates, columns=['Latitude', 'Longitude'])

# Create a Folium map centered at the mean coordinates
map_center = [df_coordinates['Latitude'].mean(), df_coordinates['Longitude'].mean()]
map_object = folium.Map(location=map_center, zoom_start=12)

# Add markers for each address
for index, row in df_coordinates.iterrows():
    folium.Marker(location=(row['Latitude'], row['Longitude'])).add_to(map_object)
```

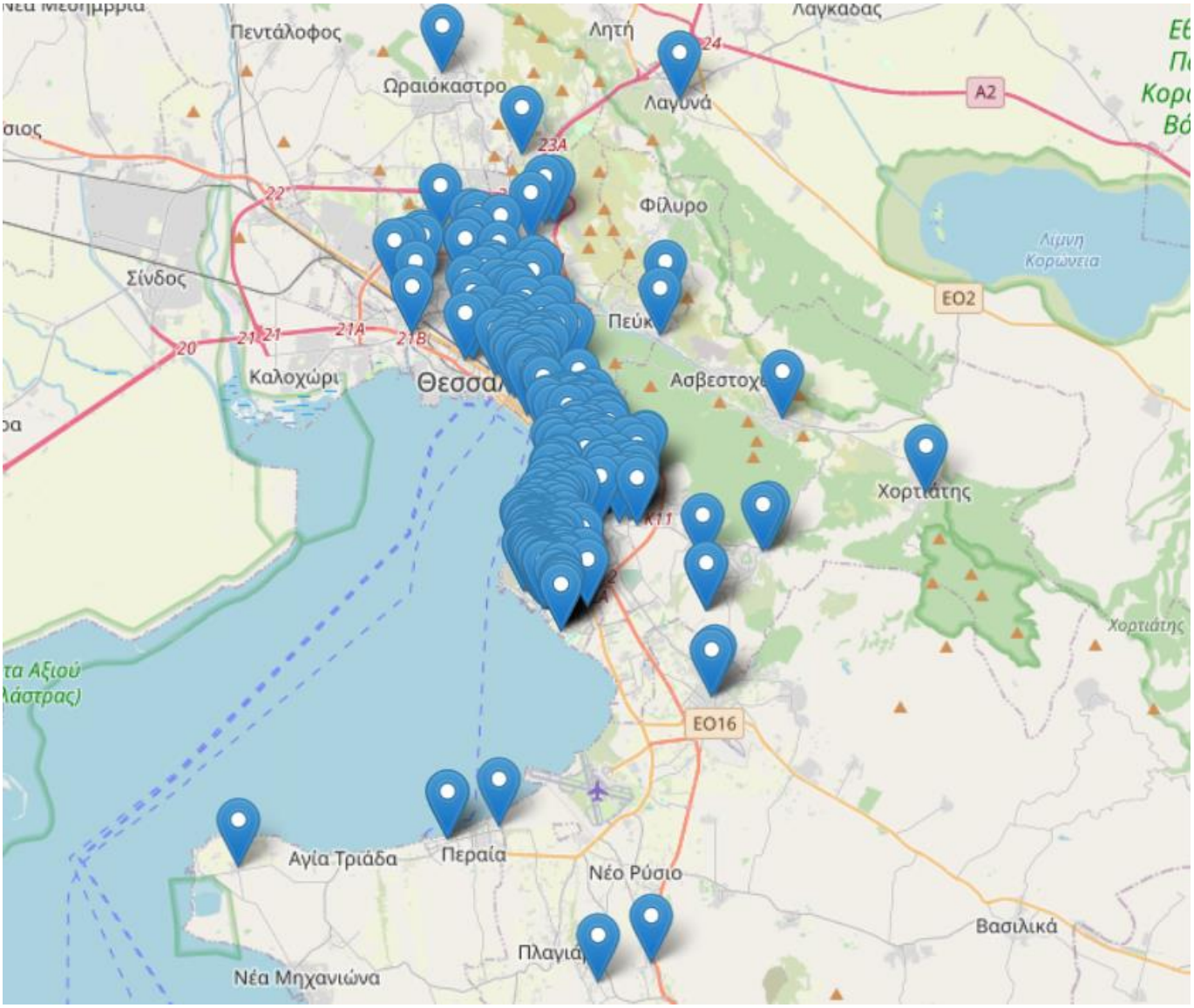


Figure 9. Thessaloniki sample visualization

Each question of the survey corresponds to a feature in the resulting dataset. Additional columns were generated for checkbox questions like 5, indicating binary values of either yes or no. For instance, the possible answers in question 5 are Car, Motorcycle, Truck or Nothing. Two columns were created: Car_licence and Motor_licence that store whether the respondent has a licence for those or not. Possession of truck licence was not considered for analysis. The same procedure was implemented also for question 6, creating two new features: Bike_access and Skate_access. It was decided that access to car and motor stores similar information to car and motor licence features, hence only access to bike and skate options were considered for analysis. Additionally, two distance-related features were generated using the Home address and Work address. More specifically, distance in kilometres was

manually computed between home and work address using Google Maps. An illustration of the procedure is presented in the figure below:

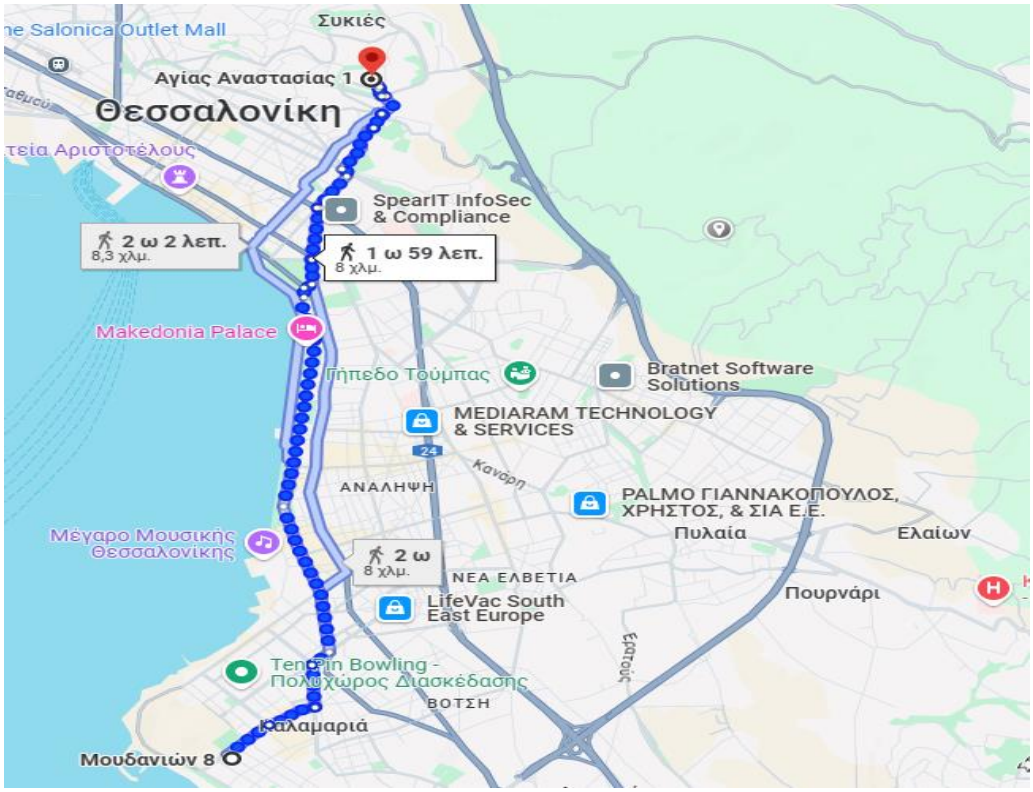


Figure 10. Google Maps Distance calculation

Also, the corresponding time for each distance was used to validate the responses from the survey in question 11 and the time (in minutes) required for commute. Specifically, respondents were requested to indicate the duration of their commute in minutes. In cases where there was a substantial difference between their provided answers and the calculated time in Google Maps, the corresponding time from Google Maps was adopted. For example, if a respondent stated a 40-minute commute, but Google Maps suggested lower than 20, the Maps estimate was used. In contrast, if a respondent implied a 30-minute commute and Google Maps indicated 33 or 34 minutes, the respondent's provided time was selected. This procedure was manually executed for all 383 samples.

The second distance-related feature was calculated in a more automated way using the Python library of Geopy. The library provides access in geocoding enabling users to find the geographic coordinates of a location based on the provided address. Based on those coordinates Geopy can calculate the distance between home and work address in a completely automated way. The code is illustrated below.

```python
# Create a geolocator instance with a unique user agent
geolocator = Nominatim(user_agent="MyGeocodingApp_Marios_Melachroinos_v2123242526272829")


# Function to get coordinates for an address
def get_coordinates(address):
    location = geolocator.geocode(address)
    if location:
        return location.latitude, location.longitude
    else:
        return None

# Function to calculate distance between home and work coordinates
def calculate_distance(row):
    home_coords = get_coordinates(row['Home_address'])
    work_coords = get_coordinates(row['Work_address'])

    if home_coords and work_coords:
        return geodesic(home_coords, work_coords).kilometers
    else:
        return None

# Apply the calculate_distance function to each row in the DataFrame
df['Geodesic_distance'] = df.apply(calculate_distance, axis=1)
```

Though, the main difference from Google Maps, is that the corresponding distance is a geodesic one. Geodesic distance is the shortest distance between two points but on a curved surface like that of the earth (Kettle, 2014). This can be better understood for distances using the airplane, where in a 2D map the shortest path between two points seems to be a straight line, but the actual shortest path is the geodesic one because of the curved surface of the earth. This example is demonstrated in the figure below with the geodesic distance represented by the blue line.

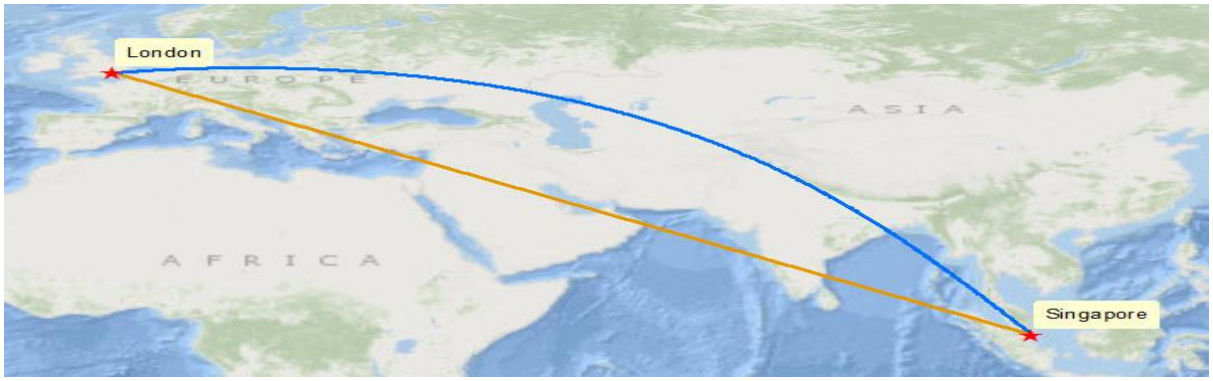


Figure 11. Geodesic Distance example

Source: Kettle, S., 2014

For the library to calculate the distance, home and work address had to be transformed into the form:

Address 99, Thessaloniki, 551 32

While many respondents provided the address in the correct form, many had to be transformed. An illustration of the transformation is depicted below.

Μουδανιών 8	Αγίας Αναστασίας 1
Παπαναστασιου 3 καλαμ	Κουντουριώτου 1 Λεωφο
Βαζελώνος 15	Θεμιστοκλή Σοφούλη 17
Χαιριανών 6	Τσιμισκή 29
Αλεξιάδη 7	Καραμανλή 46
Κρώμνης 38	ΒΙ.ΠΕ Θεσσαλονίκης Σίν
Βαζελώνος 15	Κερασούντος 69

Μουδανιών 8, Καλαμαριά 551 32	Αγίας Αναστασίας 1, Θεσσαλονίκη 546 34
Αλ. Παπαναστασίου 3, Καλαμαριά 551 32	Λεωφ. Νίκης 1, Θεσσαλονίκη 546 24
Χαιριανών 6, Καλαμαριά 551 33	Τσιμισκή 29, Θεσσαλονίκη 546 24
Αλεξιάδη Κωνσταντίνου 7, Καλαμαριά 551 32	Κων/νου Καραμανλή 46, Καλαμαριά 551 32
Κρώμνης 38, Καλαμαριά 551 31	Φλέμινγκ 7, Σίνδος 574 00
Βαζελώνος 15, Καλαμαριά 551 32	Κερασούντος 69, Καλαμαριά 551 31
Λεωφ. Ανδρέα Παπανδρέου 210, Συκιές 566 26	Κύπρου 17, Θεσσαλονίκη 546 41
Τσαλδάρη 10, π, Θέρμη 570 01	11ο χλμ εθνικής οδού, Α/Δ Θεσσαλονίκης Ν. Μουδανιών, Θεσσαλονίκη 570 01

The purpose of calculating two types of distances is to use them individually as features in classification models and evaluate whether there are significant differences in the model outcomes depending on the use of separate distance metrics. The resulting dataset comprises 383 samples and 24 features. The analysis procedure that was followed for the dataset is illustrated in the figure below:

3.1.3 Analysis procedure

The figure below illustrates the analysis procedure for the dataset. Initially, the dataset was explored for erroneous answers and outliers. Afterwards, an exploratory data analysis was executed to better understand the dataset and identify patterns. The data was the preprocessed so the models could be applied. Preprocessing includes encoding into numeric form, setting up train and test sets and normalization of data. Four models were applied: Decision Tree, Random Forest, XGBoost and a Stacking model. Different iterations were used to build the models such as feature reduction, swapping the distance metrics and parameter tuning. The model predictive performance was evaluated based on overall accuracy, recall, precision and auc score. The model with the best overall performance on those metrics was selected as the most appropriate for predictions on new unseen data.

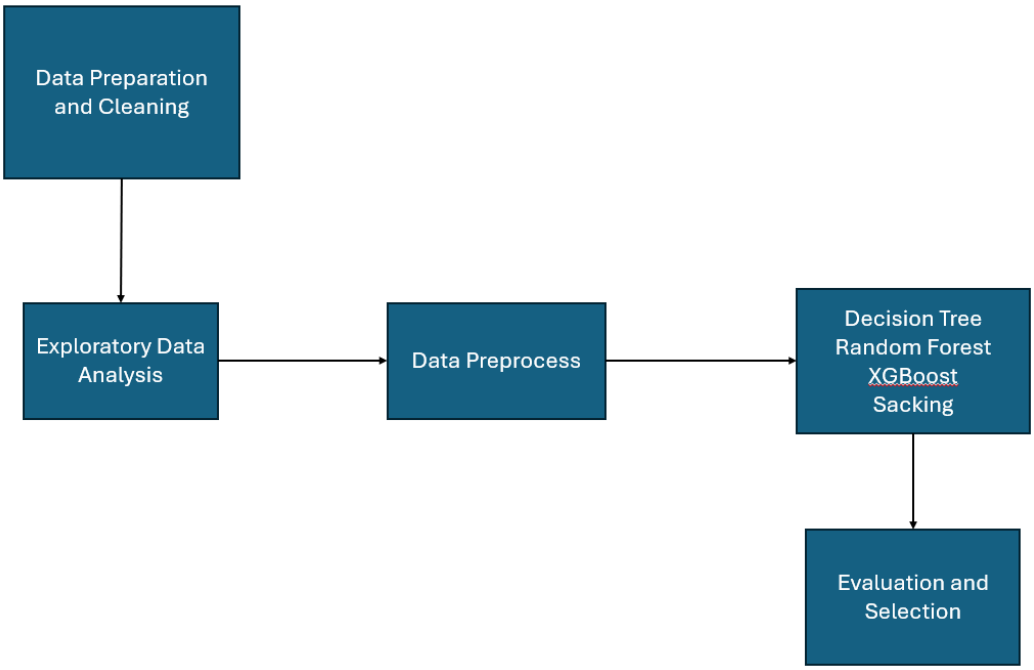


Figure 12. Analysis procedure

3.2. RESULTS

This section of the thesis will present the findings related to Thessaloniki and commuting behaviour. The initial part covers data preparation and cleaning, followed by exploratory data analysis and data preprocessing. The final segment focuses on model application, mode classification, and the selection of the best model.

3.2.1. Data preparation and cleaning

Out of the 409 samples obtained from the online survey, blank responses were excluded initially. Subsequently, responses with errors in home or work addresses (those unidentifiable in Google Maps) were eliminated. Lastly, a few samples indicated respondents working from home or concealing either of their addresses; these samples were also excluded. Consequently, from the initial 409 samples, 383 were deemed valid. Furthermore, as detailed in the methodology section, additional features were generated for the two checkbox questions, considering only "yes" or "no" values. Following the creation of these features, the data were imported into Python for further preparation and cleaning. Below are the columns of the dataset as imported into the virtual environment:

The column names, excluding those derived from checkboxes, essentially correspond to the questions posed in the online survey. For better management during the analysis phase, it was necessary to rename the columns. The features used are illustrated below.

Table 1. List of features

Columns	Type	Description
Home_address	object	Home address of the respondent
Work_address	object	Work address of the respondent
Gender	Categorical-Binary	Gender of the respondent
Age	Categorical	Age group of the respondent
Driver_licence	Categorical - Binary	Access to a driver licence?
Motor_licence	Categorical - Binary	Access to a motor licence?
Bike	Categorical - Binary	Access to a driver bike?
Skate	Categorical - Binary	Access to a skate?
Hsize	Numeric - Discrete	Household size
Vehicles	Numeric - Discrete	Number of private vehicles in household
Income	Categorical	Income group of respondents
Mode	Categorical	Mode for commuting to work (Private vehicle, bus, walk)
Time	Numeric - Continuous	Minutes required for transit to work
Depart_time	Categorical	Time of departure for commuting to work
Convenience	Categorical - Ordinal	Degree of influence of convenience (1 to 5)
Cost	Categorical - Ordinal	Degree of influence of cost (1 to 5)
Safety	Categorical - Ordinal	Degree of influence of safety (1 to 5)
Environment	Categorical - Ordinal	Degree of influence of environmental concerns (1 to 5)
Health	Categorical - Ordinal	Degree of influence of exercise and health (1 to 5)
Weather	Categorical - Ordinal	Degree of influence of weather (1 to 5)
Parking	Categorical - Ordinal	Degree of influence of parking availability (1 to 5)
Distance	Numeric - Continuous	Travel distance (kms) between home and work address

In total, there are 24 features, comprising 4 numeric, 16 categorical, and 2 object features. The next steps involved checking for null values using the **df.isnull().any()** command and identifying duplicates with **df.duplicated().sum()**. Given that blank samples were eliminated prior to importing into Python, there were no instances of nulls or duplicates.

Furthermore, erroneous samples were also removed. There was a case where respondent indicated commuting to work by private vehicle without possessing a car or motor licence. Also, it was examined whether there are instances of respondents commuting to work with a private vehicle without having any vehicles in their household, though no such answer was found.

Afterwards, a lone sample was excluded, where the respondent suggested a trip distance exceeding 80 kilometres, constituting an outlier within the dataset. The last step was to calculate the geodesic distance using the geopy library in python. The code is illustrated below.

```
# Create a geolocator instance with a unique user agent
geolocator = Nominatim(user_agent="MyGeocodingApp_Marios_Melachroinos_v2123242526")


# Function to get coordinates for an address
def get_coordinates(address):
    location = geolocator.geocode(address)
    if location:
        return location.latitude, location.longitude
    else:
        return None

# Function to calculate distance between home and work coordinates
def calculate_distance(row):
    home_coords = get_coordinates(row['Home_address'])
    work_coords = get_coordinates(row['Work_address'])

    if home_coords and work_coords:
        return geodesic(home_coords, work_coords).kilometers
    else:
        return None

# Apply the calculate_distance function to each row in the DataFrame
df['Geodesic_distance'] = df.apply(calculate_distance, axis=1)
```

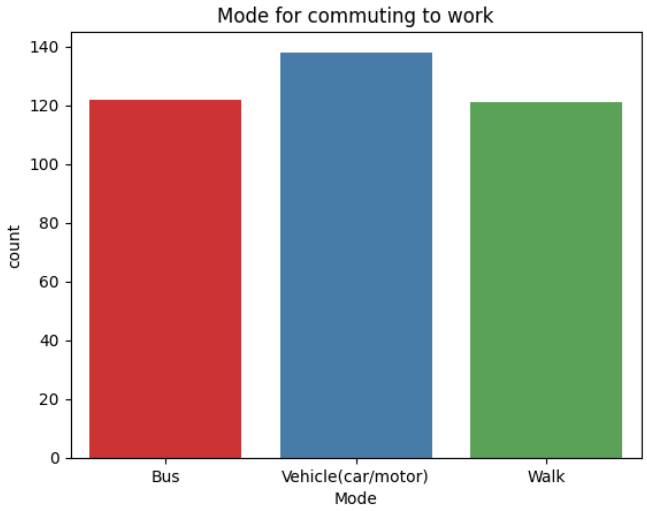
There were samples where the process could not identify distances between addresses, so NaN values returned for those cases as shown below.

Distance	Geodesic_distance
8.00	7.157182
7.60	13.814558
6.60	6.063515
0.35	NaN

Those NaN values were replaced with the corresponding Distance (that was calculated manually). For instance, the above NaN value was replaced with 0.35. The same procedure was followed for all instances with NaN values. Consequently, 381 valid options were retained to proceed with the exploratory data analysis phase. The final dataset consists of 381 samples and 21 features.

3.2.2. Exploratory Data Analysis

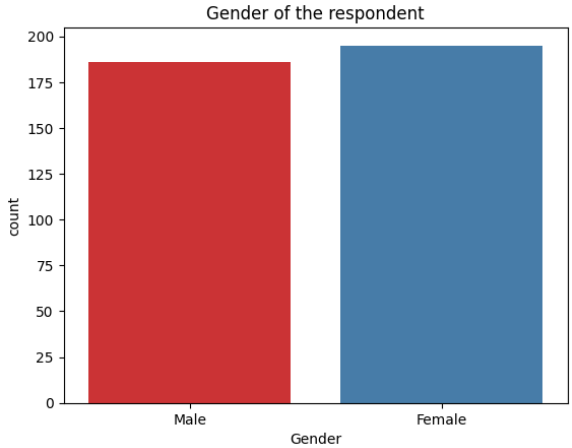
In this section of the Thesis the descriptive statistics along with visualisation for each feature will be presented to better understand the dataset. The first feature is about the target variable and the commuting mode to work for the respondents. It is evident that there is a balance among private vehicle, walk and bus users with over 100 observations for each. Note that private vehicle includes both commuters via car and motorcycle. Displayed below is the count plot of the feature, accompanied by the corresponding percentages for each mode.



Mode	Percentage
Bus	32.02%
Walk	31.75%
Vehicle (car/motor)	36.22%

Figure 13 Transport Mode

The following feature refers to the gender of the respondents. It is apparent that there is a relatively even distribution among the responses. To be more specific, there are slightly over 175 observations for both men and women, with women having a slim numerical edge (51.18%). The grouped plot also illustrates that a higher proportion of women use the bus compared to men. In contrast, men tend to use private vehicles slightly more than women.



Gender	Percentage
Women	51.18 %
Men	48.82 %

Figure 14. Gender

The subsequent set of features pertains to whether the respondent possesses driving licences. The options for car and motor licences only allow for "yes" or "no" responses. The plots make it apparent that most respondents possess a car licence compared to motor licence users.

	Driver licence (car)	Motor licence
Yes	**68,24 %**	**84.25 %**
No	**31.75 %**	**15.74 %**

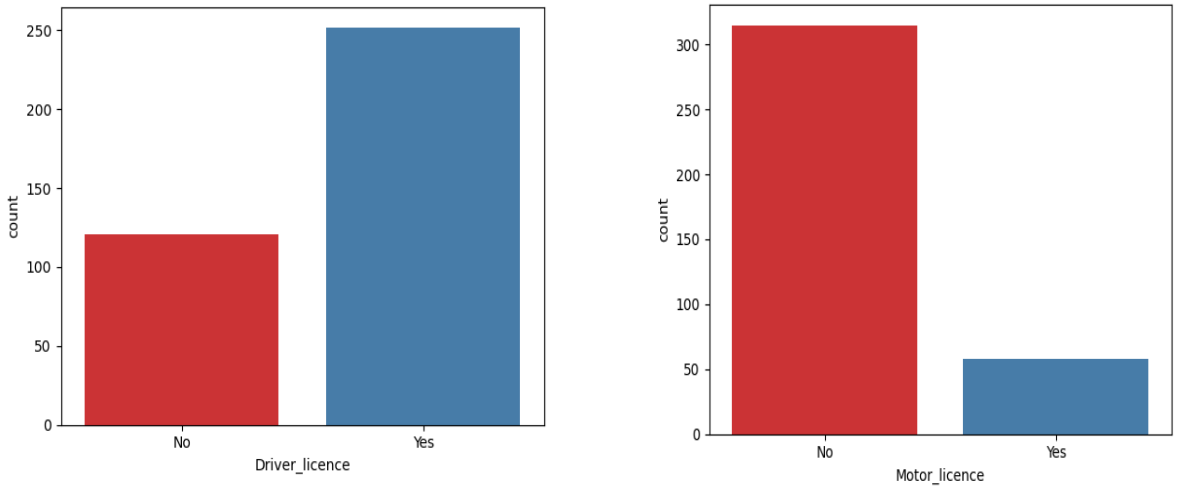


Figure 15. Driver and Motor licence

The following group of features pertains to whether the respondent has access to bike, or skate. Like the preceding set, these features only permit "yes" or "no" as possible answers. In both situation most of the respondents imply not possessing any of those.

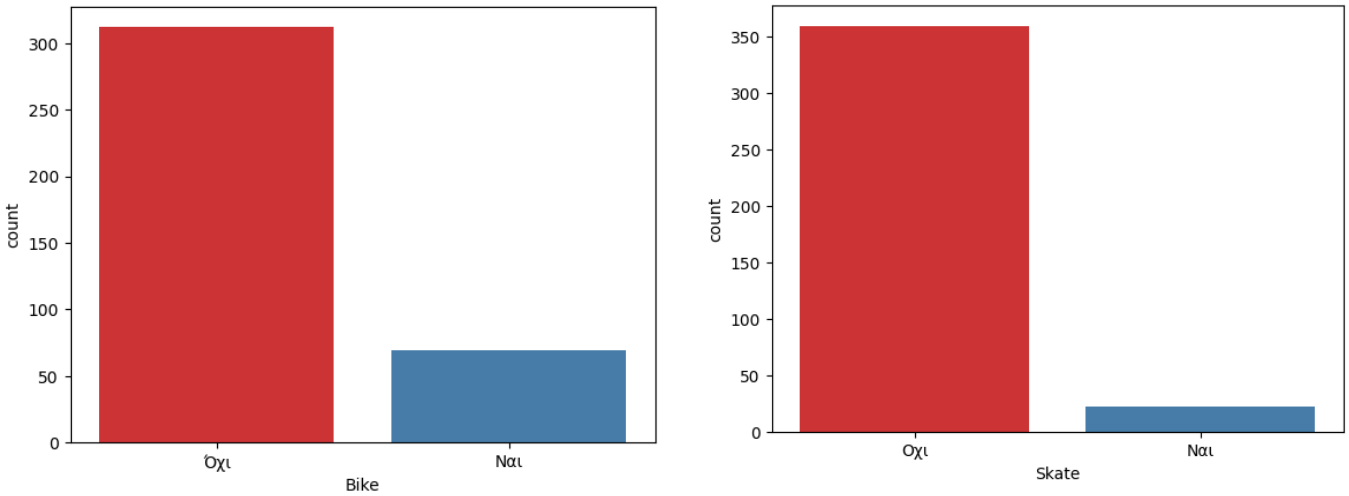
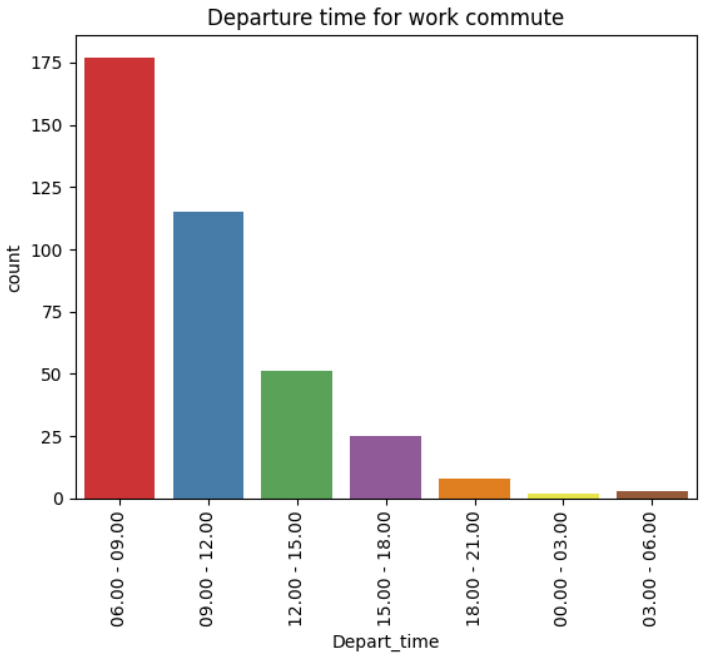


Figure 16. Skate and Bike access

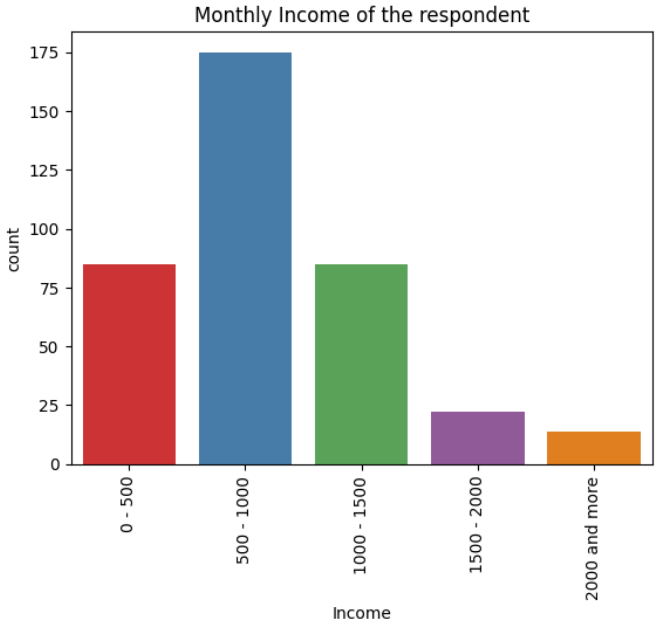
The next feature concerns the departure time of the respondent for their work commute. As depicted in the plot below, it is apparent that most respondents commute to work during the morning hours between 06:00 and 12:00. There are also a few instances where respondents commute during the evening hours between 12:00 and 18:00, while the minority proportion commutes to work at off hours during 18:00 - 21.00 and 00.00 - 06.00 past midnight. There are no instances where respondents depart between 21.00 and 00.00.



Depart time	Percentage
06.00-09.00	46.45 %
09.00-12.00	30.18 %
12.00-15.00	13.39 %
15.00-18.00	6.56 %
18.00-21.00	2.09 %
00.00-03.00	0.78 %
03.00-06.00	0.52 %

Figure 17. Departure time

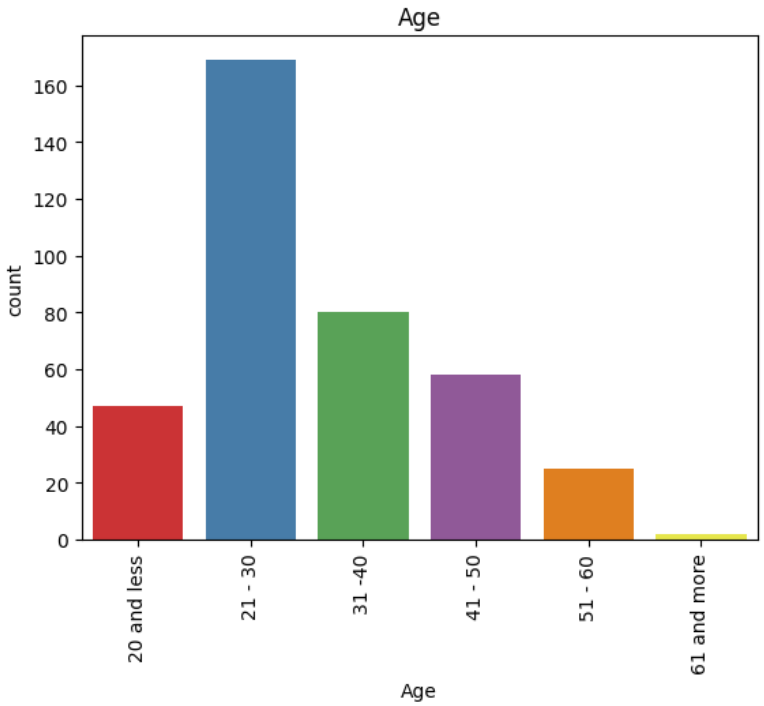
The following question refers to the monthly income of the respondent. As depicted in the graph, most respondents have a monthly income between 500 and 1000£, followed by those with incomes between 0 and 500£. Additionally, there is a substantial proportion of respondents with an income exceeding the 1000£ mark, with a few observations even surpassing 2000£.



Income	Percentage
0 - 500	22.30 %
500 - 1000	46.93 %
1000 - 1500	22.30%
1500 - 2000	5.74 %
2000 and more	3.67 %

Figure 18. Income

The next feature pertains to the age group of the respondent. The vast majority fall within the 21 to 30 years old category, followed by those aged between 31 and 40. For the remaining age groups, the observations are relatively more balanced. The sole exception is the age group of 61 and above, which has fewer than 5 observations.



Age	Percentage
Below 20	12.33 %
21 - 30	44.35%
31 - 40	20.99 %
41 - 50	15.23 %
51 - 60	6.56 %
More than 61	0.52%

Figure 19. Age

The following feature is about the household size of the respondent. The overwhelming majority indicates residing in households comprising between 2 and 5 members. In addition to this, a minority of respondents suggests living in larger families with 6 or more members, while a small proportion mention being part of a single-member family.

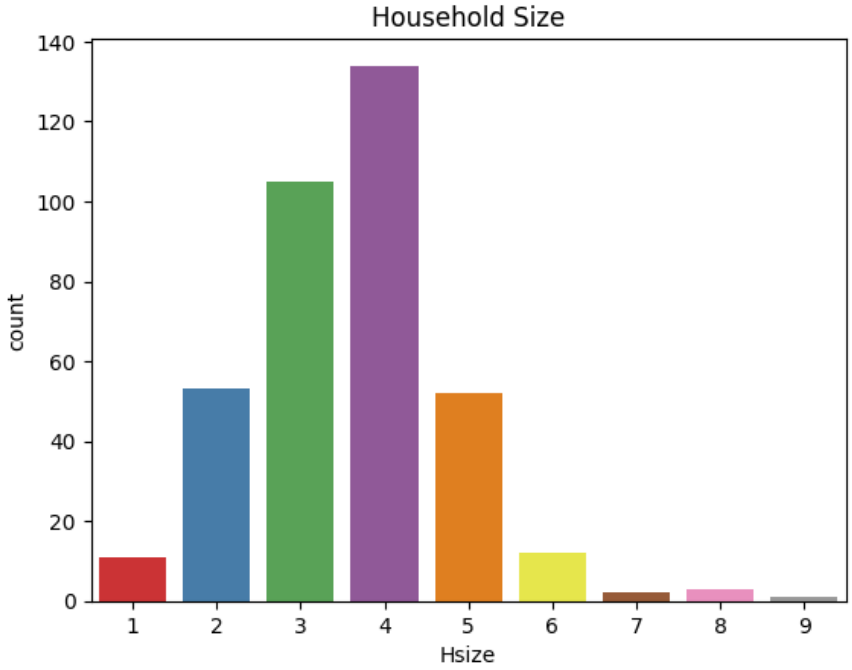
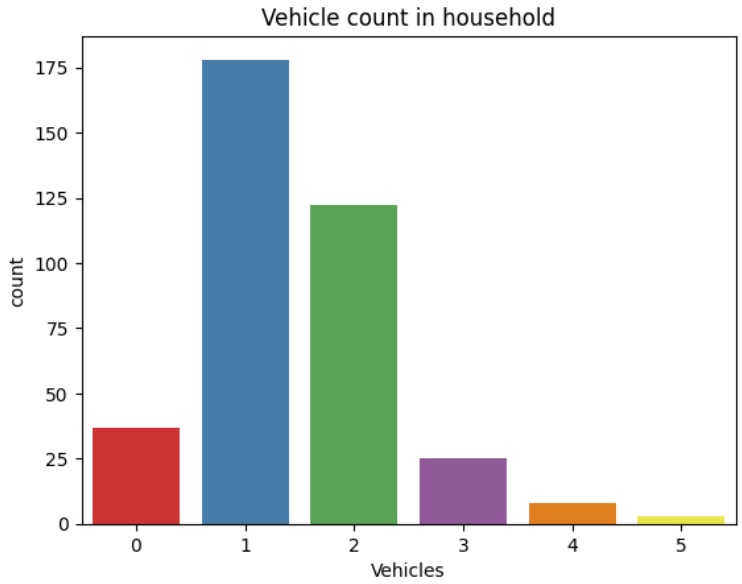


Figure 20. Household Size

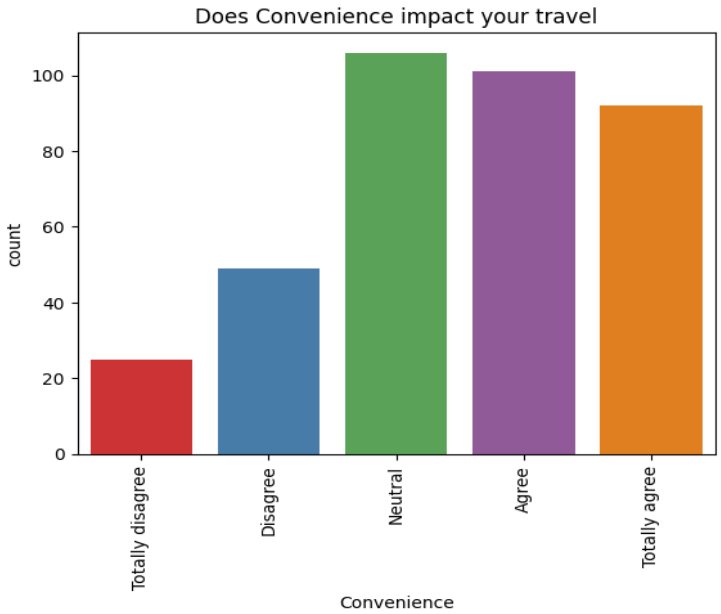
The subsequent feature concerns the number of vehicles in the households of the respondents. Most respondents suggest living in a household with 1 or 2 private vehicles. Additionally, a minority of respondents mentions residing in a house with no vehicles at all, while a few observations suggest households with 4 or 5 vehicles.



Number of vehicles	Percentage
0	9.92 %
1	47.72 %
2	32.71 %
3	6.70 %
4	2.14 %
5	0.80 %

Figure 21. Number of vehicles

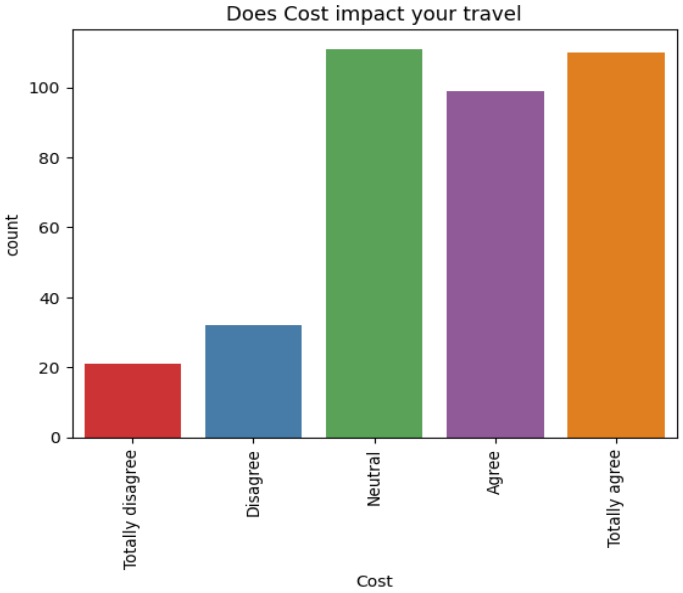
The next series of questions addresses the impact of specific factors on the commuting behaviour of the respondent. Starting with convenience as an influencing factor, a significant number of respondents expressed either "agree" or "totally agree" that the convenience of transportation affects their commuting behaviour to work. Additionally, there is a substantial proportion of respondents indicating a "neutral" response to the impact of convenience. Fewer observations are categorised as "disagree" or "totally disagree."



Impact of Convenience	Percentage
Totally Disagree	6.56 %
Disagree	12.86 %
Neutral	28.34 %
Agree	27.29 %
Totally Agree	24.93 %

Figure 22. Convenience

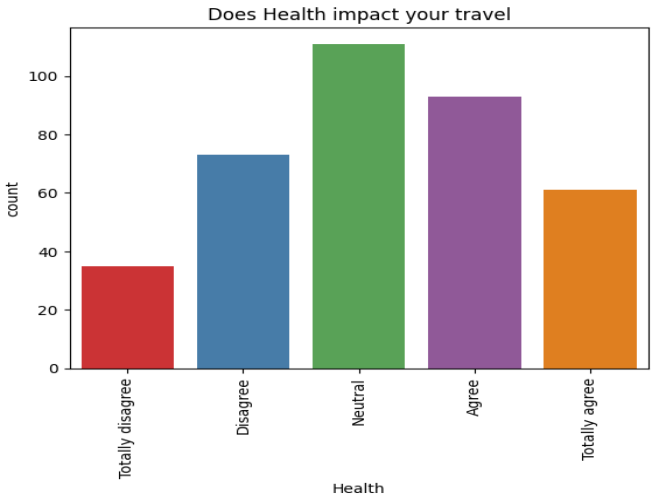
The next feature refers to the impact of transportation cost in commuting behaviour. Most of the respondents also expressed “Agree” and “Totally Agree” that cost is a significant factor influencing their transits to work. Additionally, a significant portion expressed a "neutral" position regarding this aspect. In contrast, only a small number of responses suggested that cost does not impact their commuting behaviour, as reflected in the "Disagree" and "Totally Disagree" categories.



Impact of Cost	Percentage
Totally Disagree	5.51 %
Disagree	8.39 %
Neutral	29.92 %
Agree	25.98 %
Totally Agree	30.18 %

Figure 23. Cost

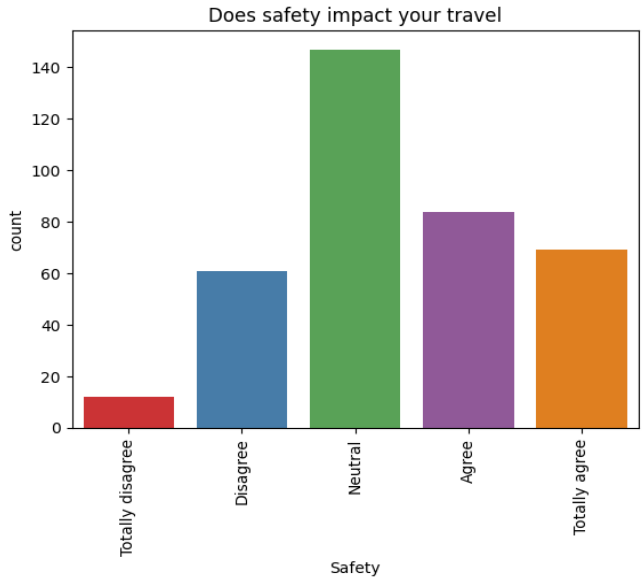
The following factor is about physical exercise and health. Specifically, it refers to how physical exercise such as walking and consequently health concerns in general affect the commuter’s transit to work. The results are quite balanced with respondents implying both negative and positive categories. Also, a significant number of respondents also indicated a “neutral” position regarding health impact.



Impact of Health	Percentage
Totally Disagree	9.44 %
Disagree	19.42 %
Neutral	29.92 %
Agree	24.93 %
Totally Agree	16.27 %

Figure 24. Physical activity and health

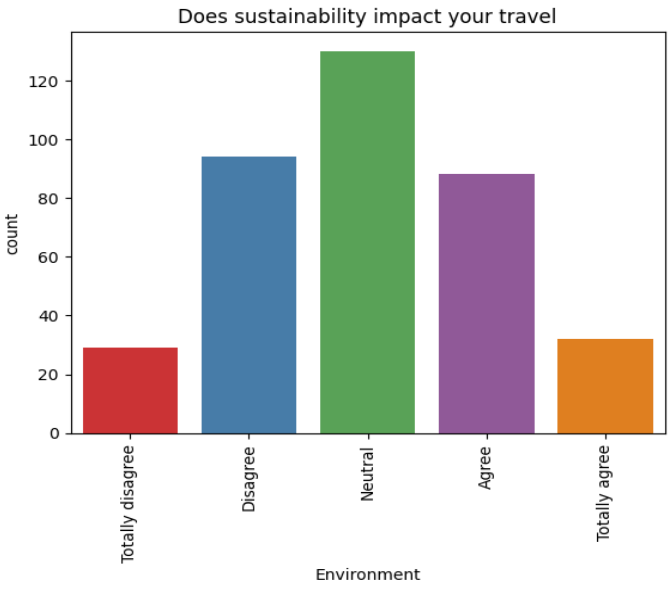
The next feature pertains to the factor of transportation safety. Once again, most respondents express a neutral stance regarding the impact of safety on their commuting behaviour to work. Additionally, a significant proportion reacts positively to the importance of this factor. A smaller number of respondents indicate that safety does not influence their transit behaviour.



Impact of Safety	Percentage
Totally Disagree	3.41 %
Disagree	16.01 %
Neutral	39.10 %
Agree	22.57 %
Totally Agree	18,89 %

Figure 25. Safety

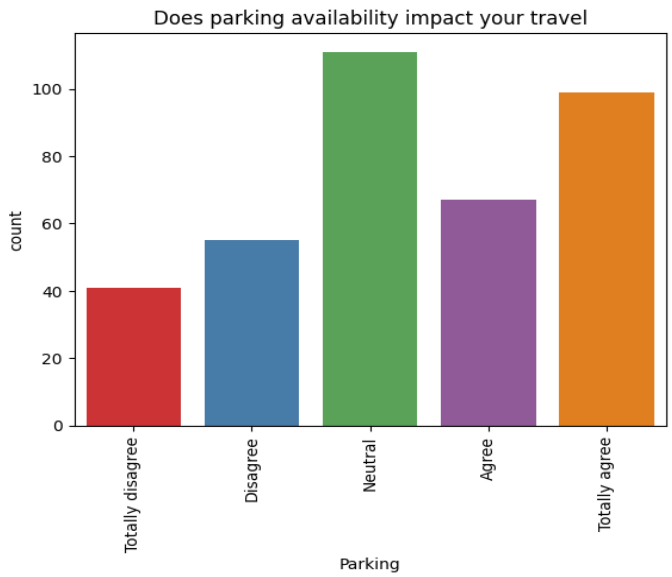
The subsequent feature focuses on the influence of environmental concerns. In a manner akin to health and physical exercise, respondents exhibit a balanced stance, with most observations equally distributed across positive and negative categories. Once more, many responses align with a neutral stance for this specific factor.



Impact of Environmental concerns	Percentage
Totally Disagree	7.87 %
Disagree	24.93%
Neutral	34.64 %
Agree	23.88 %
Totally Agree	8.66 %

Figure 26. Environmental concerns

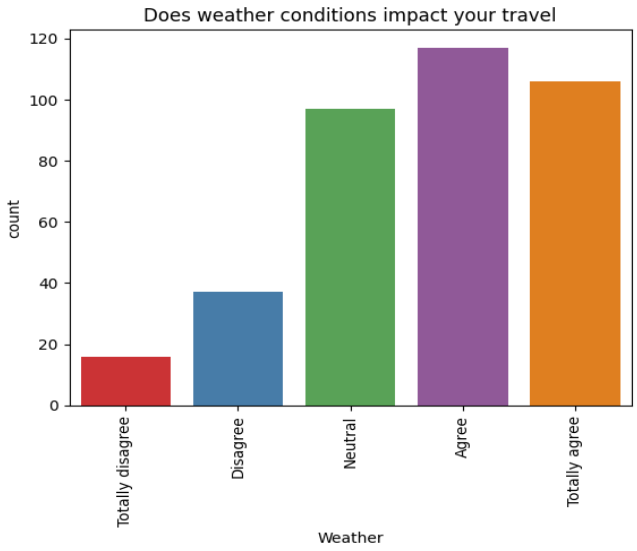
Transitioning to the next factor, it concerns the impact of parking availability on commuting behaviour to work. In this context, numerous observations fall into the positive categories, particularly the "totally agree" option, with the "neutral" category being the second most common. The negative categories encompass fewer observations regarding the impact of this factor.



Impact of parking availability	Percentage
Totally Disagree	11.02 %
Disagree	14.69 %
Neutral	29.39 %
Agree	18.37 %
Totally Agree	26.50 %

Figure 27. Parking availability

The final factor concerns the impact of weather conditions on commuting to work. In this aspect, there is a notable contrast between positive and negative responses, with most respondents falling into the "agree" and "totally agree" categories. A substantial proportion also indicated a neutral stance on the factor, while both the "disagree" and "totally disagree" categories represent vast minorities.



Impact of weather	Percentage
Totally Disagree	4.29 %
Disagree	14.74 %
Neutral	29.76 %
Agree	17.96 %
Totally Agree	26.54 %

Figure 28. Weather impact

The subsequent set of features pertains to the distance metrics calculated between the home and work addresses. The "Distance" feature signifies the optimal travel distance manually obtained from Google Maps, whereas the "Geodesic Distance" represents the distance automatically calculated by the Geopy library in Python. Both distances are expressed in kilometres. The histograms and boxplots for both metrics, along with their descriptive statistics, are depicted below.

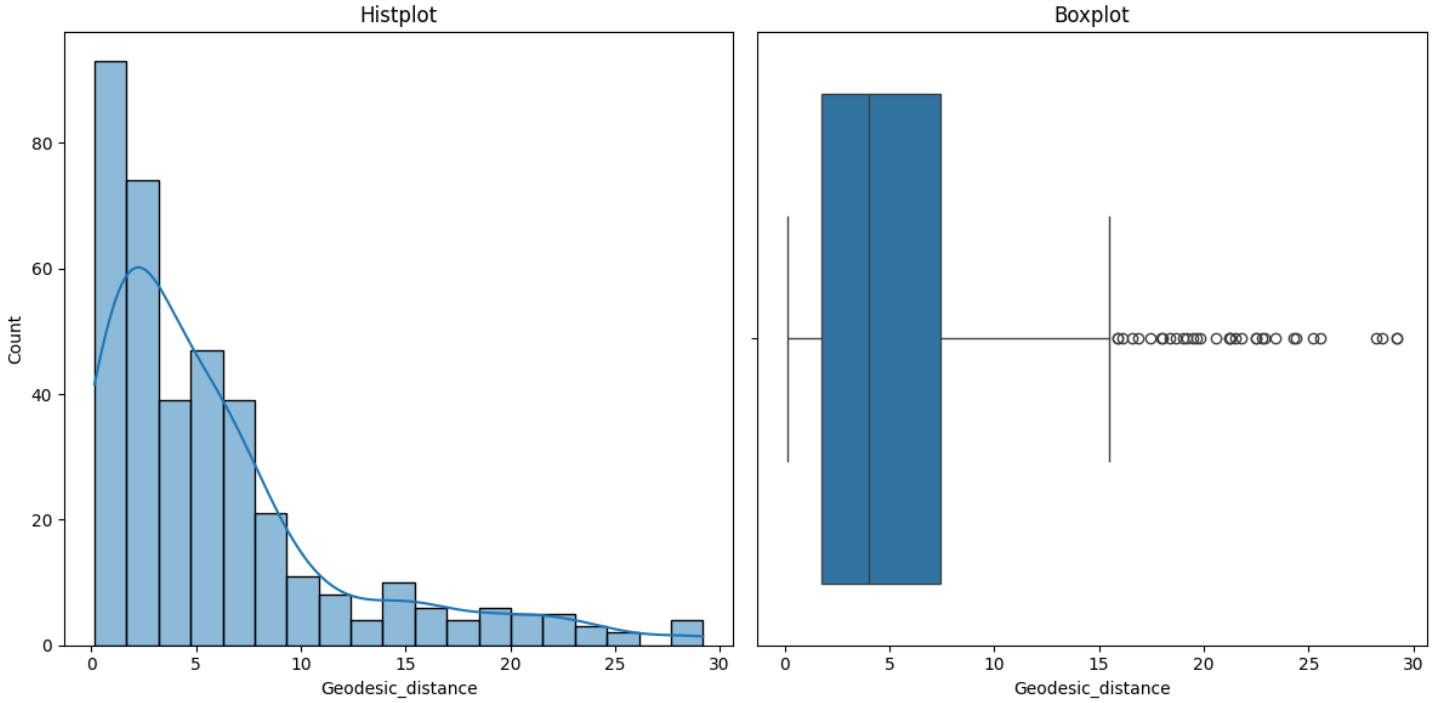


Figure 29. Geodesic Distance plots

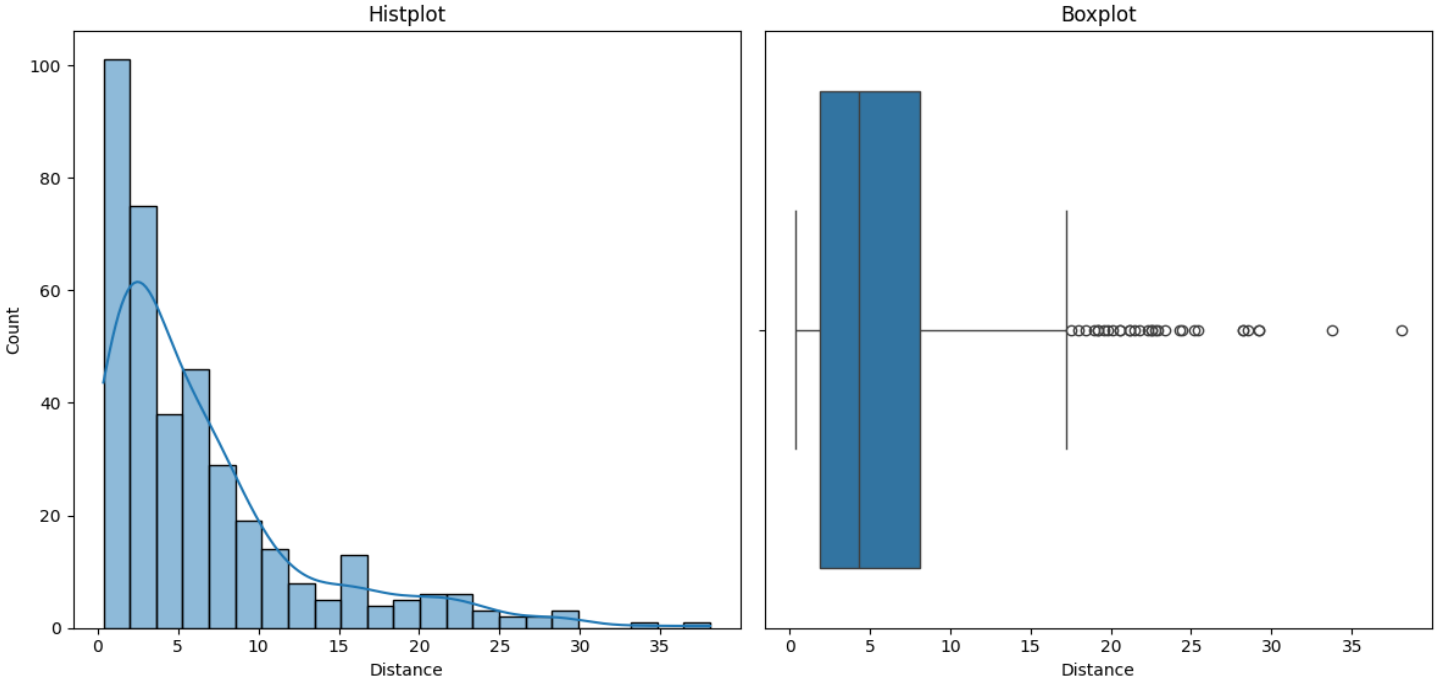


Figure 30. Distance plots

Table 2. Distance and Geodesic Distance descriptives

	Distance	Geodesic Distance
Mean	**6.48**	**5.97**
std	**6.51**	**6.01**
min	**0.35**	**0.16**
25 %	**1.9**	**1.73**
50 %	**4.3**	**4.04**
75 %	**8.1**	**7.41**
max	**38.1**	**29.2**

The graphs reveal that most observations are concentrated in the initial classes, ranging from 0.1 to 8 kilometres, regardless of the distance metric calculated. Furthermore, both distributions exhibit positive skewness while all the descriptive statistics for geodesic distance appear slightly lower when compared to optimal travel distance. The graphs below show the density of each mode for both distance metrics. Density represents the concentration of data points for a continuous variable in a normalized version over count plots. As illustrated on the plots below the concentration of data for walk is mostly within 0 to 5 kilometres while for bus is within 0 and 10 kilometres. For private vehicle the distribution is more spread across 0 and 35 kilometres.

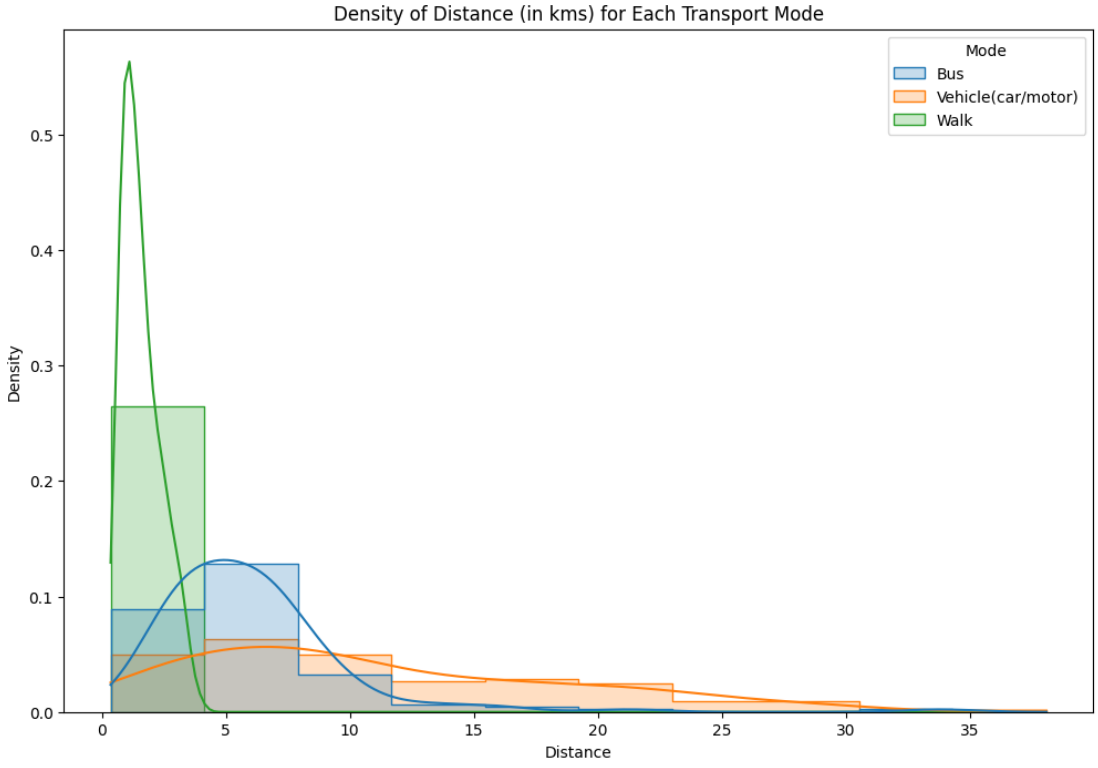


Figure 31 Density of Distance by mode

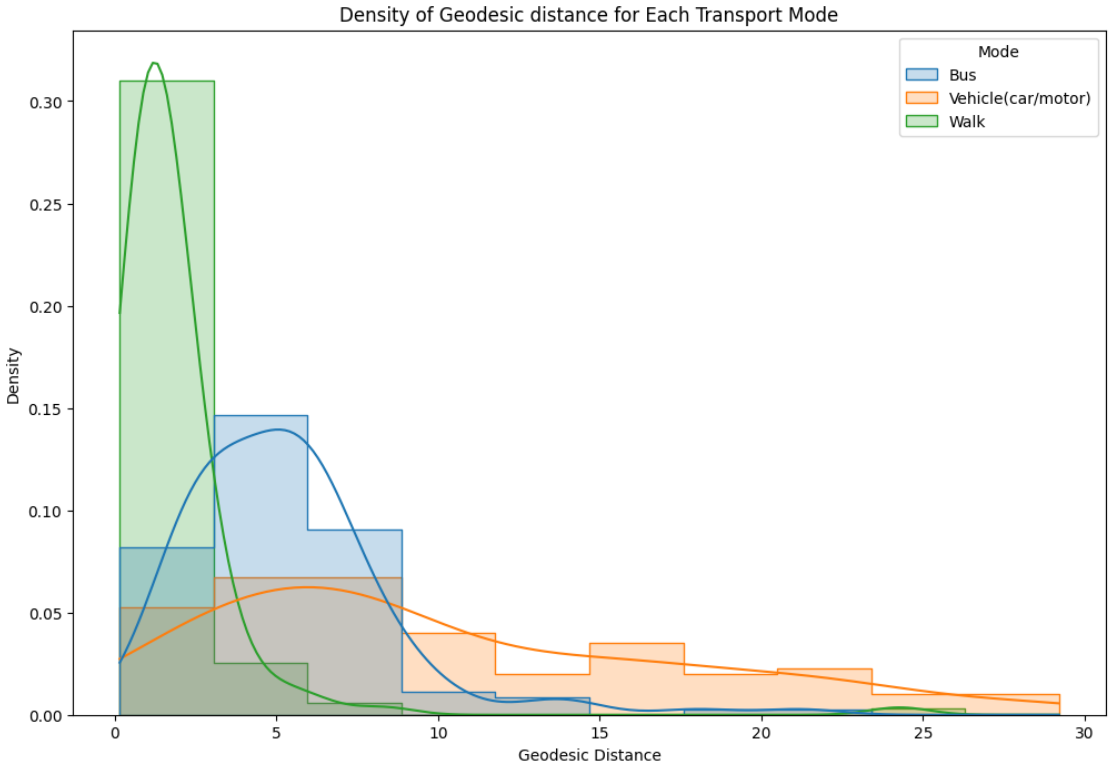


Figure 32. Density of Geodesic Distance by mode

The final feature concerns the time required (in minutes) for respondents to commute to work, measured in minutes. It can be observed that it follows a distribution like the distance features, indicating a positive skew, with many commutes occurring in the initial classes of approximately 3 to 45 minutes. The histogram, along with the descriptive statistics, are depicted below.

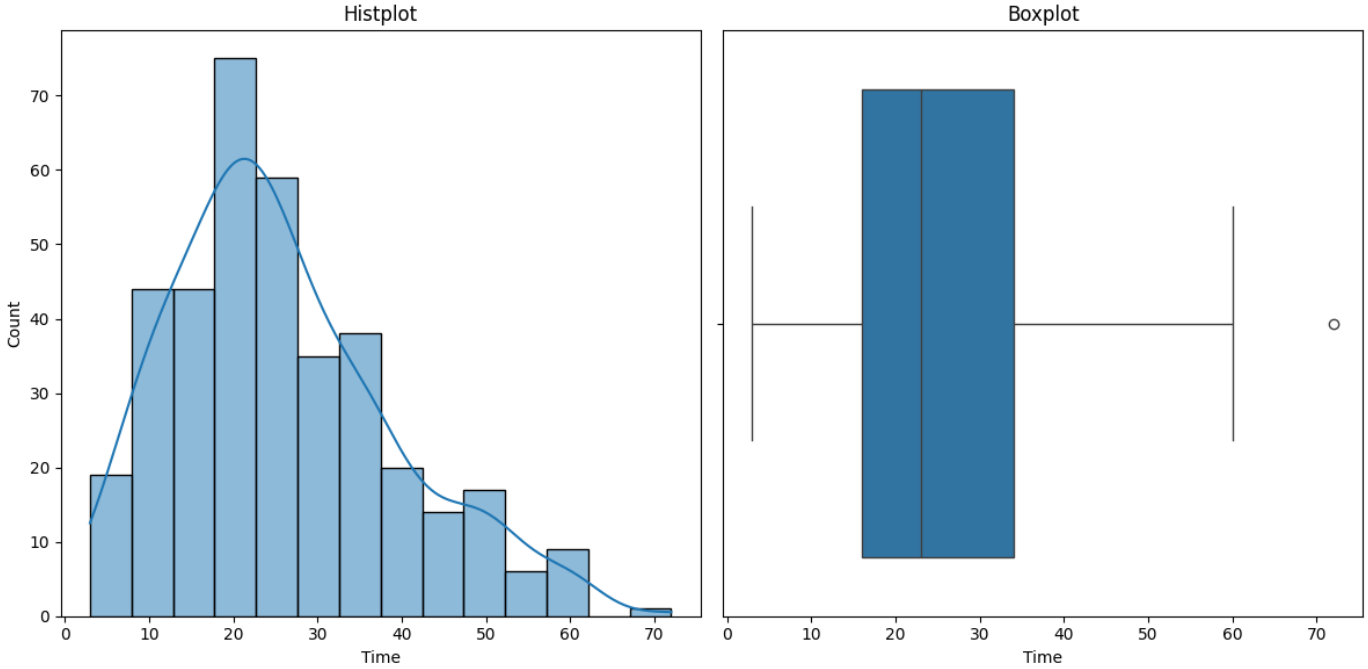
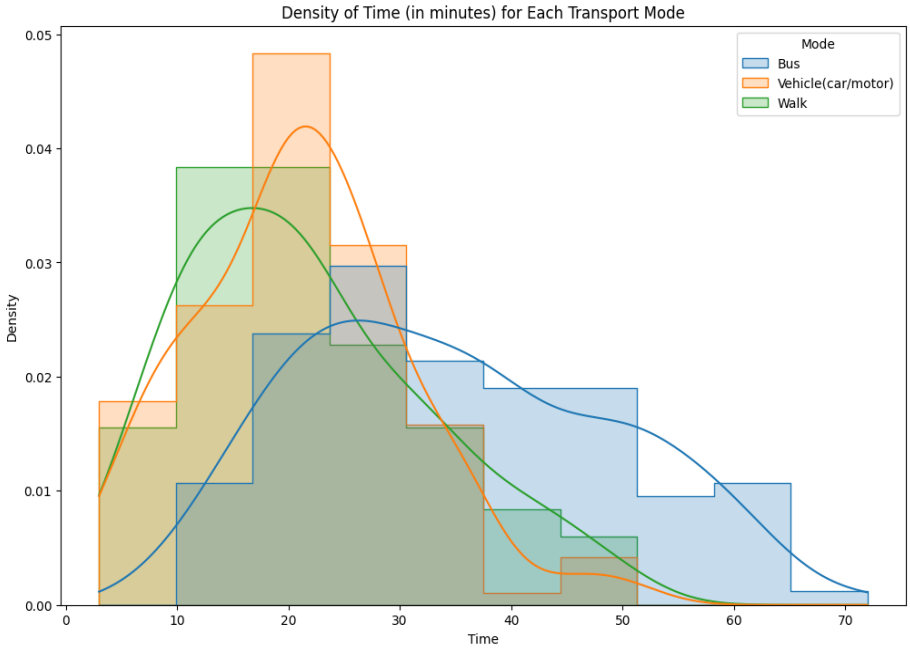


Figure 33. Duration (in minutes)



	Time
Mean	25.78
std	13.3
min	3
25 %	16
50 %	23
75 %	34
max	72

Figure 34. Density of duration and descriptives

The density plot was again constructed for time and each transport mode. For bus density is more spread across the X-axis while walking and private vehicles are more concentrated within 5- and 40-minutes commute time.

Additional graphs were also generated, to better understand the data. The following graph illustrates the departure time grouped by transport mode. At "peak" hour, typically between 06.00 and 09.00 most of the observations are for private vehicles. Though, as departure time progresses bus and walk commuters overcome those who commute with private vehicles, specifically during 12.00 and 15.00.

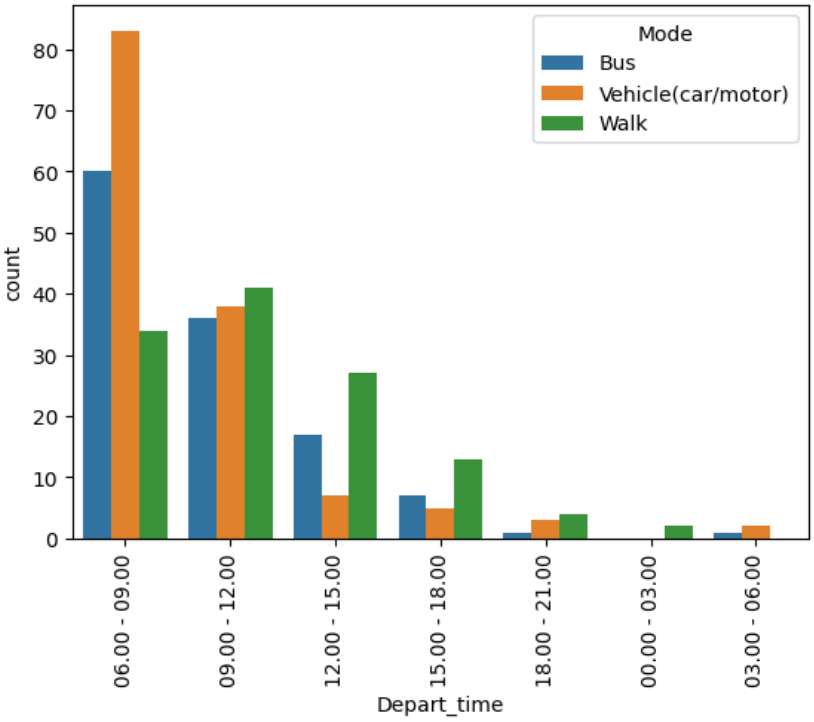


Figure 35. Departure time by mode

The subsequent graph displays the age of the respondents categorized by transport mode. It is apparent that most respondents below 20 years old refrain from commuting to work using a private vehicle. In contrast, as age increases, there is a tendency for individuals to commute to work using a private vehicle. Fewer people opt for bus or walking as their mode of commuting beyond the age of 41.

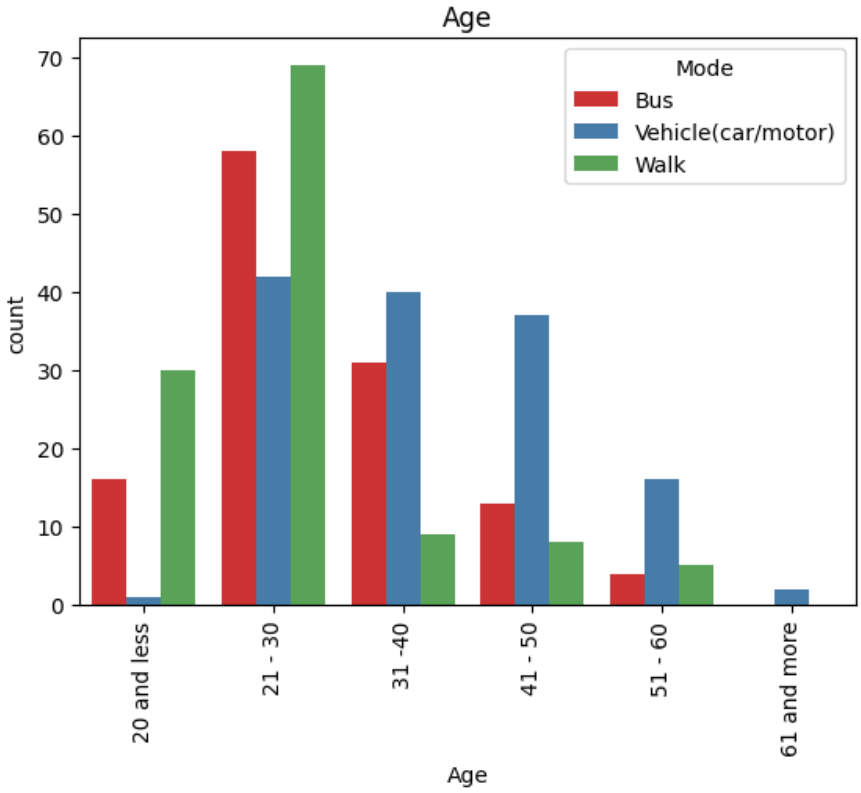


Figure 36. Age by mode

The following graph illustrates the income range grouped by transport mode. The trend is likewise the age of the respondent. When income falls within the 0 to 500 range, commuters predominantly opt for either bus or walking. Conversely, as income increases, private vehicles become the apparent choice for commuting.

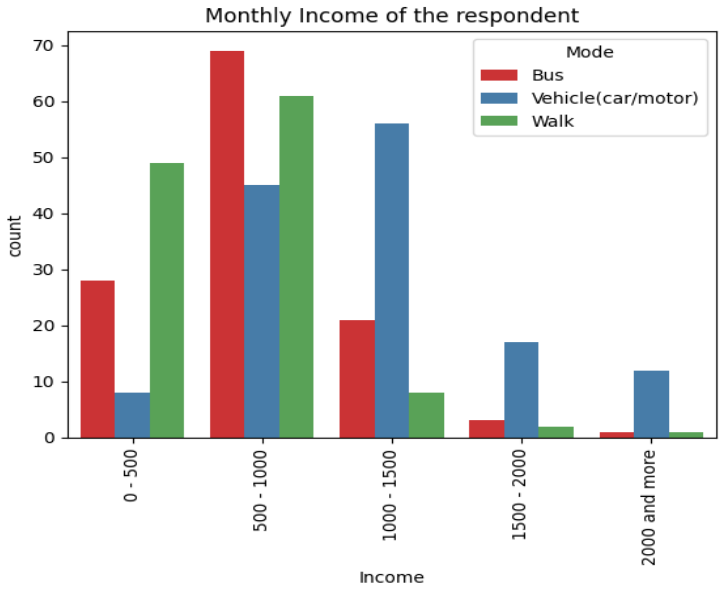


Figure 37. Income by mode

3.2.3. Data Preprocess

To prepare the data for the classification model, a preprocessing step is necessary to ensure the models function effectively. It is crucial to encode all features into numerical form, as models can only process numerical data. Firstly, low frequencies in some categories were grouped so to not lose information from the dataset. More specifically for Age and Depart time low frequencies were grouped as depicted below.

```
perc=df['Age'].value_counts(True)*100
print(perc)
```

```
Age
21 - 30        44.356955
31 -40         20.997375
41 - 50        15.223097
20 and less    12.335958
51 - 60         6.561680
61 and more     0.524934
Name: proportion, dtype: float64
```

```
value_mapping = {"51 - 60":'51 and more', "61 and more":'51 and more'}
df['Age'] = df['Age'].replace(value_mapping)
```

```
perc=df['Age'].value_counts(True)*100
print(perc)
```

```
Age
21 - 30        44.356955
31 -40         20.997375
41 - 50        15.223097
20 and less    12.335958
51 and more     7.086614
Name: proportion, dtype: float64
```

```
value_mapping = {"15.00 - 18.00":'Other', "18.00 - 21.00":'Other', "03.00 - 06.00":"Other", "00.00 - 03.00":"Other"}
df['Depart_time'] = df['Depart_time'].replace(value_mapping)
```

```
perc=df['Depart_time'].value_counts(True)*100
print(perc)
```

```
Depart_time
06.00 - 09.00    46.456693
09.00 - 12.00    30.183727
12.00 - 15.00    13.385827
Other             9.973753
Name: proportion, dtype: float64
```

For Age “51 – 60” and “more than 61” responses were grouped into one category, “51 and more. For Depart Time responses outside of the 06.00 - 15.00 range were grouped into “Other” category. The Income feature was numerically encoded by selecting the mean of each range as its value. For instance, the value 250 was assigned for the income range "0-500," 750 for "500-1000," and so forth. The encoding for Income is illustrated below.

```
value_mapping = {"0 - 500":250, "500 - 1000":750, "1000 - 1500": 1250, "1500 - 2000":1750, "2000 and more":225
df['Income'] = df['Income'].replace(value_mapping)
```

```
perc=df['Income'].value_counts(True)*100
print(perc)
```

```
Income
750     45.931759
250     22.309711
1250    22.309711
1750     5.774278
2250     3.674541
```

Additionally, the impact factors (cost, convenience, safety, health, environment, parking, weather) were encoded into an ordinal scale of 1-5. More specifically the encoding is illustrated below:

```
value_mapping = {"Totally disagree":1, "Disagree":2, "Neutral":3, "Agree":4, "Totally agree":5}
df['Convenience'] = df['Convenience'].replace(value_mapping)
```

```
value_mapping = {"Totally disagree":1, "Disagree":2, "Neutral":3, "Agree":4, "Totally agree":5}
df['Cost'] = df['Cost'].replace(value_mapping)
```

```
value_mapping = {"Totally disagree":1, "Disagree":2, "Neutral":3, "Agree":4, "Totally agree":5}
df['Safety'] = df['Safety'].replace(value_mapping)
```

```
value_mapping = {"Totally disagree":1, "Disagree":2, "Neutral":3, "Agree":4, "Totally agree":5}
df['Health'] = df['Health'].replace(value_mapping)
```

```
value_mapping = {"Totally disagree":1, "Disagree":2, "Neutral":3, "Agree":4, "Totally agree":5}
df['Weather'] = df['Weather'].replace(value_mapping)
```

```
value_mapping = {"Totally disagree":1, "Disagree":2, "Neutral":3, "Agree":4, "Totally agree":5}
df['Environment'] = df['Environment'].replace(value_mapping)
```

```
value_mapping = {"Totally disagree":1, "Disagree":2, "Neutral":3, "Agree":4, "Totally agree":5}
df['Parking'] = df['Parking'].replace(value_mapping)
```

Furthermore, all the binary features were encoded into 0 and 1. This process essentially assigns 0 for "No" and 1 for "Yes" observations. Gender is also part of the label encoder process with 0 assigned to "Males" and 1 to "Females".

The last features that require encoding are Age and Depart_time. For those features one-hot encoding process was used. This procedure essentially generates new binary features for each potential answer, assigning values of 0 to indicate the absence of that value in the specific row and 1 to imply the presence of that value in the row. The process is illustrated below.

```
one_hot_encoded = pd.get_dummies(df[['Age', 'Depart_time']])
one_hot_encoded = one_hot_encoded.astype(int)
df = pd.concat([df, one_hot_encoded], axis=1)
```

An example of the results from the one hot encoding process is depicted below.

Age_20 and less	Age_21 - 30	Age_31 -40	Age_41 - 50	Age_51 and more	Depart_time_06.00 - 09.00	Depart_time_09.00 - 12.00
0	1	0	0	0	0	1
0	1	0	0	0	1	0
0	1	0	0	0	1	0
0	1	0	0	0	1	0
0	1	0	0	0	1	0

The next step was to display the correlation matrix. It is crucial to examine correlations among various features to determine which ones to drop prior to the modelling application. These decisions rely on the correlation coefficient values, which can range from -1 to 1. Values of 1 and -1 imply a perfect correlation (positive or negative) between variables, potentially leading to multicollinearity issues that could hinder the performance of the models. Hence, highly correlated features should be eliminated before the application process.

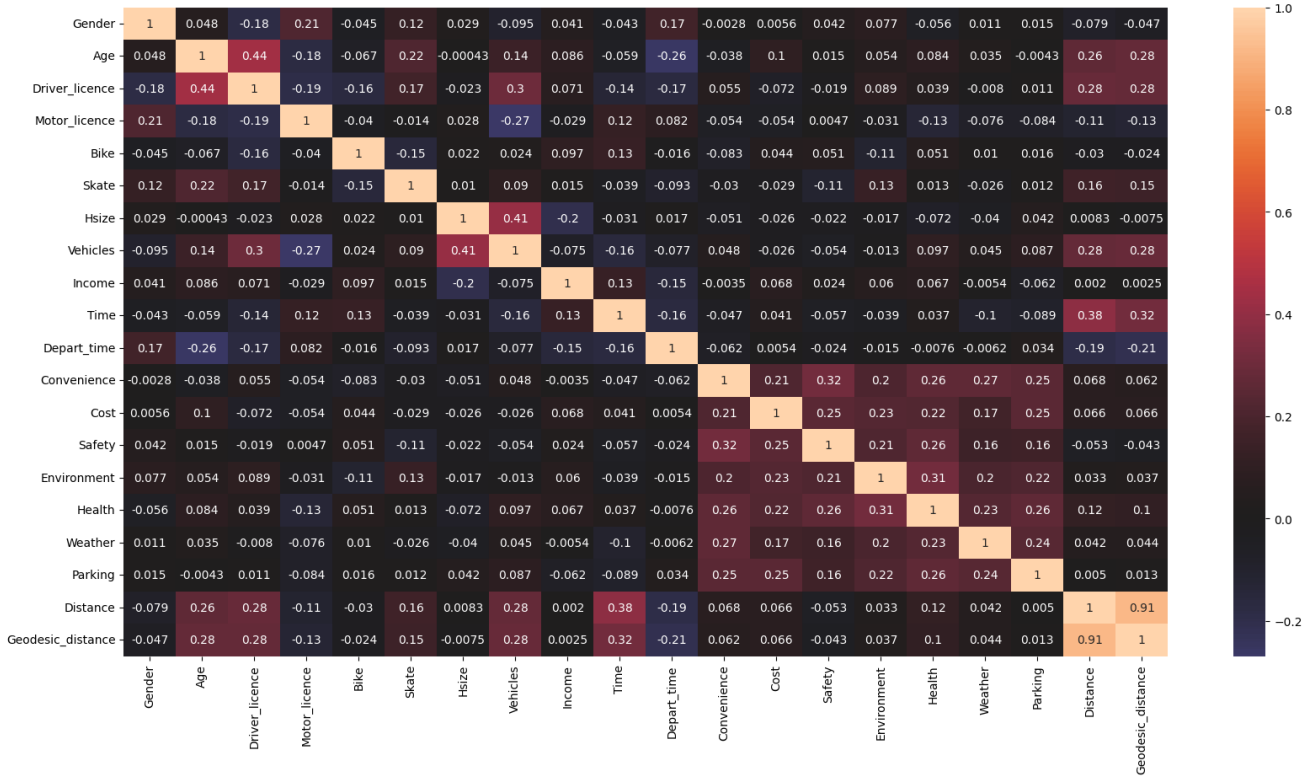


Figure 38. Correlation matrix

From the figure above, it is identified that there are strong correlations in a few cases. Distance and Geodesic_distance with a value of 0.91, followed by Driver_licence and Age with 0.44. Only one the Distance metrics will be used for training the models each time. While the correlation matrix is valuable for identifying correlated features, the Cramer's V correlation matrix was also computed to explore the association between categorical variables. The values of the correlation range from 0 to 1 with values close to 1 implying strong association among the features (Zach, 2021). The new correlation matrix is illustrated below.

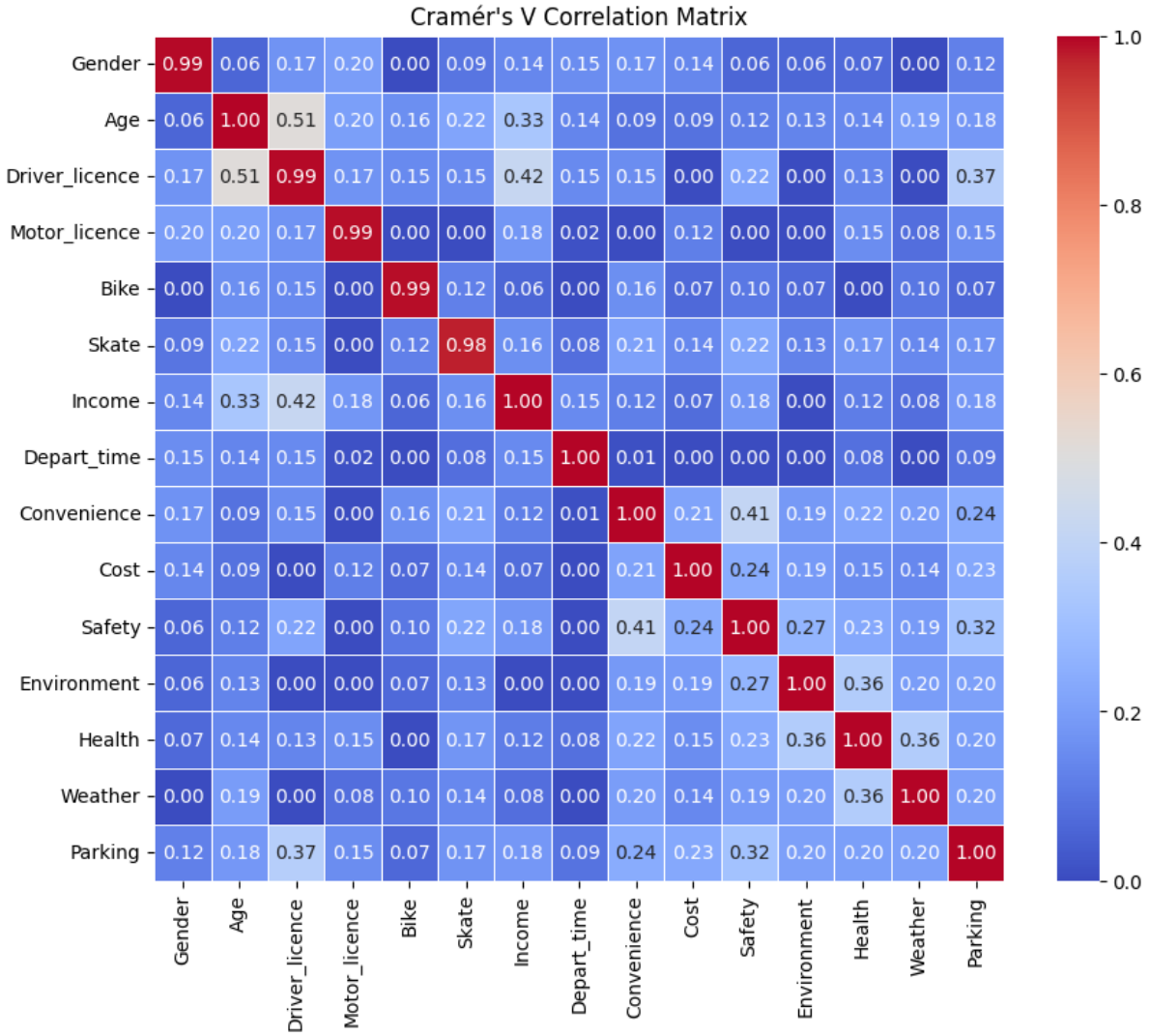


Figure 39. Cramer's V correlation

The strongest association occurs between Driver license and Age with a value of 0.51. The next highest association occurs between income and Driver license with a value of 0.42. Hence, Driver license feature will be excluded from the models.

The "vehicles" feature will also be excluded because, when it takes a value of 0 (indicating no household vehicles), it introduces a bias in the classifier, making it easier to classify non-private vehicle observations. Skate, Bike and Motor licence features will also be excluded from the models for the same reason. Additionally, to review multicollinearity issues the VIF (Variance Inflator Factor) function was

used. VIF measures the severity of multicollinearity, so each feature is assigned a value. Values above 10 raise concerns about the correlation of the features and should be removed. The results of the first attempt using VIF from statsmodels library are depicted below.

	Variable	VIF
0	const	0.000000
1	Gender	1.129359
2	Hsize	1.049693
3	Income	1.681202
4	Time	1.557271
5	Convenience	1.508518
6	Cost	1.198751
7	Safety	1.883755
8	Environment	1.513632
9	Health	1.732697
10	Weather	1.446991
11	Parking	1.470916
12	Distance	1.490938
13	Age_20 and less	inf
14	Age_21 - 30	inf
15	Age_31 -40	inf
16	Age_41 - 50	inf
17	Age_51 and more	inf
18	Depart_time_06.00 - 09.00	inf
19	Depart_time_09.00 - 12.00	inf
20	Depart_time_12.00 - 15.00	inf
21	Depart_time_Other	inf

Figure 40. VIF 1st try

The occurrence of "inf" values in certain features points to multicollinearity issues, particularly in one hot encoded features. This arises because, during the one-hot encoding process, one of the resulting binary features must be omitted to prevent multicollinearity. Consequently, depart time 06:00-09.00, and Age 21-30 were excluded. The VIF function was subsequently reapplied, yielding the following values.

	Variable	VIF
0	const	63.846924
1	Gender	1.129359
2	Hsize	1.049693
3	Income	1.681202
4	Time	1.557271
5	Convenience	1.508518
6	Cost	1.198751
7	Safety	1.883755
8	Environment	1.513632
9	Health	1.732697
10	Weather	1.446991
11	Parking	1.470916
12	Distance	1.490938
13	Age_20 and less	1.261247
14	Age_31 -40	1.432278
15	Age_41 - 50	1.420269
16	Age_51 and more	1.303088
17	Depart_time_09.00 - 12.00	1.322222
18	Depart_time_12.00 - 15.00	1.352682
19	Depart_time_Other	1.273932

Figure 41. VIF 2nd try

Now, with all VIF values below 5, these finalized features are set to be utilized in the models. The dataset was initially shuffled and consequently split into training and testing sets. The train_test_split() function of sklean was used with a ratio of 60:40. Hence, 60% of the data (228 samples) will train the models, while 40% (153) will be used for evaluation. The data was subsequently normalized using the MinMaxScaler() function from sklearn, which transforms the data values into a 0 – 1 range. The dataframes before and after the normalization process are depicted below.

Income	Time	Convenience	Cost	Safety	Environment	Health	Weather	Parking	Distance	Age_20 and less
750	26	4	3	4	4	4	3	5	21.8	0
2250	13	5	1	5	5	3	5	5	4.0	0
750	31	3	4	3	3	3	3	2	4.6	0
250	19	3	3	3	4	4	4	3	1.3	1
750	6	4	5	5	5	5	5	1	0.8	0

Income	Time	Convenience	Cost	Safety	Environment	Health	Weather	Parking	Distance	Age_20 and less
0.25	0.403509	0.75	0.50	0.75	0.75	0.75	0.50	1.00	0.568212	0.0
1.00	0.175439	1.00	0.00	1.00	1.00	0.50	1.00	1.00	0.096689	0.0
0.25	0.491228	0.50	0.75	0.50	0.50	0.50	0.50	0.25	0.112583	0.0
0.00	0.280702	0.50	0.50	0.50	0.75	0.75	0.75	0.50	0.025166	1.0
0.25	0.052632	0.75	1.00	1.00	1.00	1.00	1.00	0.00	0.011921	0.0

Figure 42. Normalization

3.2.4. Decision Tree

The first model that was applied was that of the decision tree. The DecisionTreeClassifier(random_state = 21) function was used to create the model. The random_state parameter controls the randomness and ensures reproducibility in the results. The model was fitted on the training set using the fit() command and consequently evaluated on the test set using the predict() command. The results are illustrated below.

Table 3. Decision tree performance with default parameters

	Precision	Recall	F1	Accuracy
Private Vehicle	0.73	0.78	0.75	0.78
Bus	0.83	0.69	0.76	
Walk	0.79	0.86	0.82	
Macro Average	0.78	0.78	0.78	

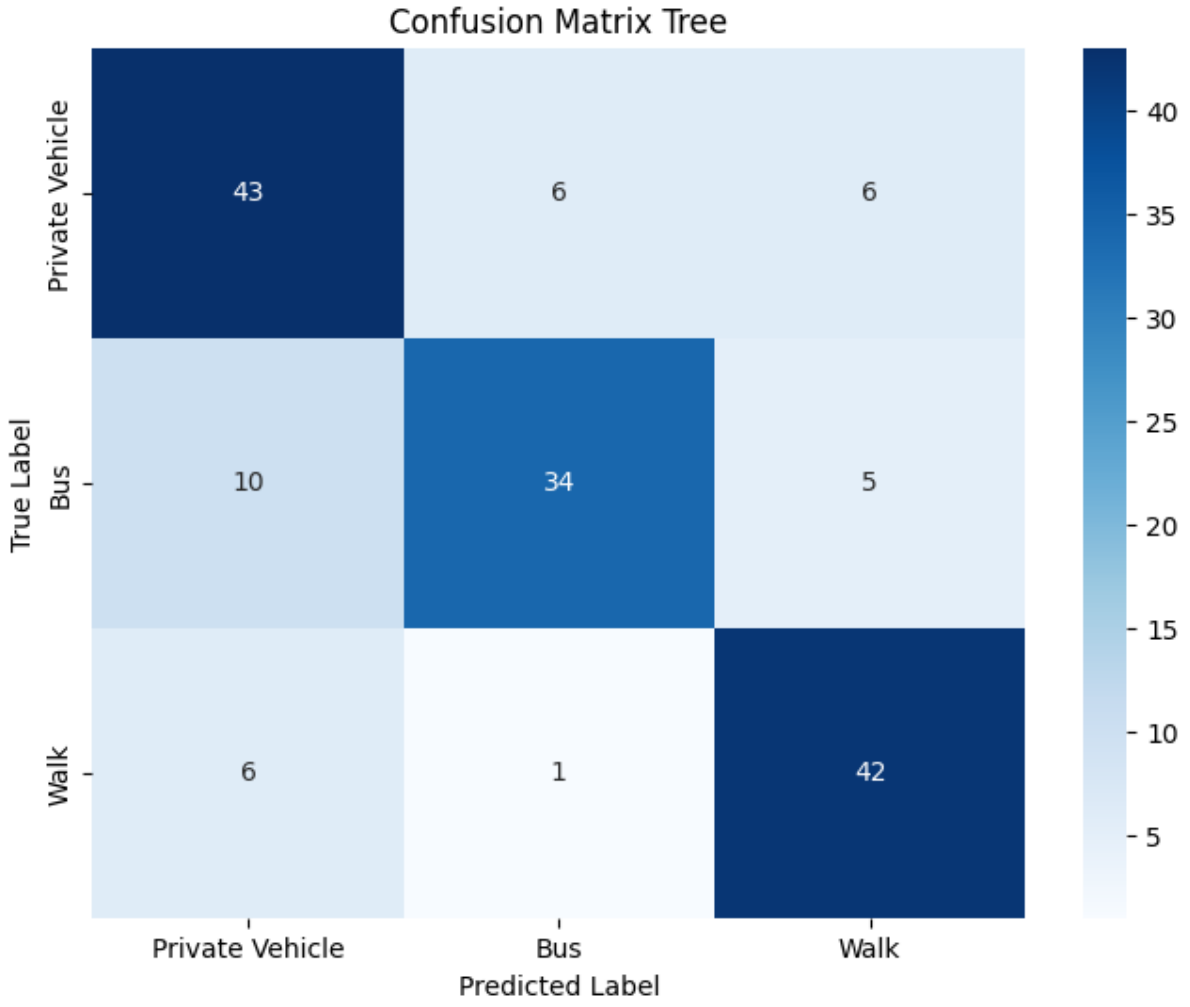


Figure 43. Confusion Matrix - Default Tree

The overall accuracy of the model is 0.78 while the max_depth of the tree was found to be 6. The max depth parameter controls the maximum possible length of the path from the root node to a leaf node. The model correctly predicted 42 out of 49 (0.86) instances as “walk” when the true label was “walk”. For classes “private vehicle” and “bus” recall was found 0.78 and 0.69 respectively. The highest precision is for class “bus” (0.83) indicating that of the 41 predictions the model made as “bus” 34 belonged in that class. The biggest misclassification occurs between private vehicle - bus, where the model predicted 10 instances as “private vehicle” while the true label was “bus”. The model clearly overfits the data since accuracy for the training set is 1, while the performance on the test set was average. To address this concern, a 10-fold cross-validation was implemented to assess whether tree pruning, which involves reducing the size of the tree, enhances the model's performance. The results of cross validation are presented below.

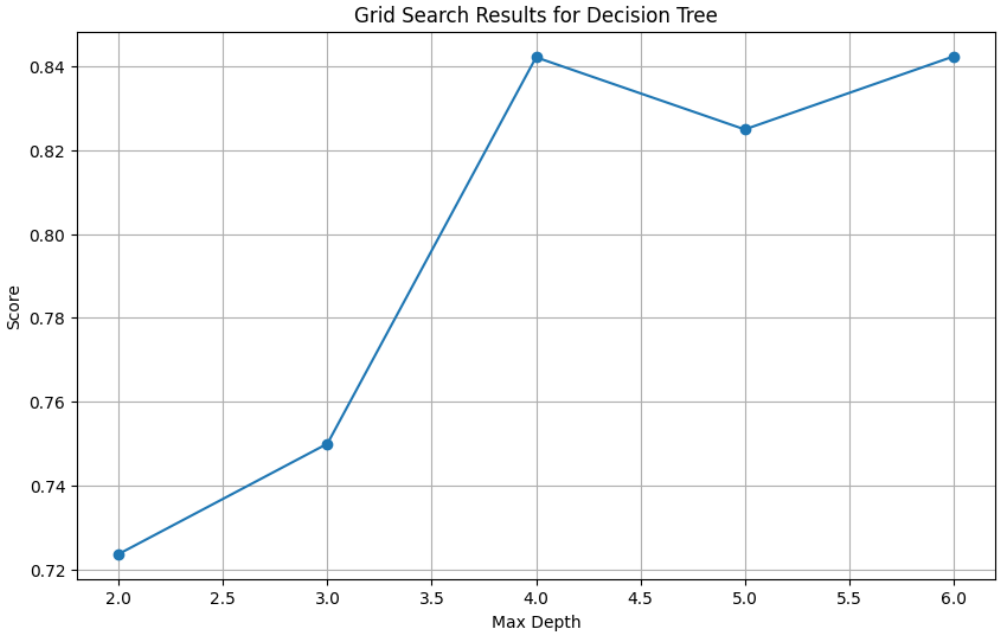


Figure 44. Cross Validation for tree depth

From the validation results it is identified that accuracy is higher for depth equal to 4 and 6. The tree was pruned at max depth value of 4 and was revaluated on the test set. The results are illustrated on the classification report below.

Table 4. Pruned Tree performance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.84	0.76	0.80	0.82
Bus	0.89	0.82	0.85	
Walk	0.76	0.90	0.82	
Macro Average	0.83	0.83	0.82	

Compared to the original tree model, the overall accuracy has been increased from 0.78 to 0.82. Recalls for both bus and walk increased. More particularly, for bus recall increased from 0.69 to 0.82, while for walk it increased from 0.86 to 0.90. In contrast, recall for private vehicle decreased from 0.78 to 0.76 with a trade off in precision as it increased from 0.73 to 0.84. Overall reducing the max depth parameter slightly increased the performance, though training accuracy remains high at 0.95. Hence, additional parameters were tuned for the model. More specifically, min_samples_split and min_samples_leaf parameters were also tuned via GridSearchCV() function. Min_samples_split refers to the number of samples required to split an internal node. In contrast, min_samples_split refers to the number of samples required to be in a leaf mode. The set of parameters for the validation process are depicted below:

```
dt_model = DecisionTreeClassifier(random_state=21)
param_grid = {
    'max_depth': [2, 3, 4, 5],
    'min_samples_split': [1, 2, 3, 4,  5, 6, 7, 8, 9, 10],
    'min_samples_leaf': [1, 2, 3, 4, 5, 6, 7],
}
```

The validation process wielded the following results:

max_depth = 5, **min_samples_split = 2**, **min_samples_leaf = 3 and a validation accuracy of 0.85**. The best model was then retrieved and evaluated on the test set. The results are depicted below.

Table 5. Tuned Tree performance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.82	0.82	0.82	0.83
Bus	0.89	0.80	0.84	
Walk	0.80	0.88	0.83	
Macro Average	0.83	0.83	0.83	

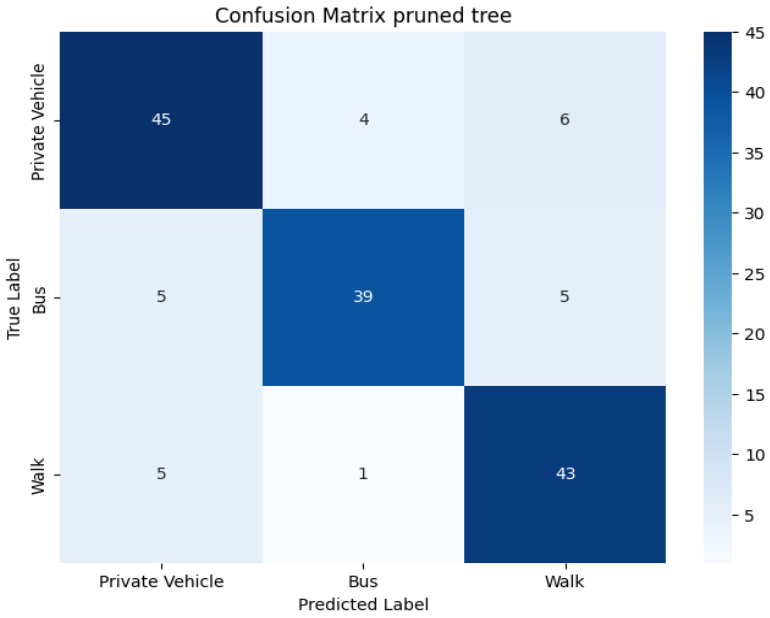


Figure 45. Confusion Matrix - Tuned Tree

The overall accuracy of the decision tree increased from 0.82 to 0.83. Simultaneously, the training accuracy remained the same at 0.95. Compared to the previous tree, recall for "bus" decreased from 0.82 to 0.80, while for private vehicle increased from 0.76 to 0.82. Furthermore, the precision for "walk" also increased from 0.76 to 0.80. The most significant misclassification occurred for walk and private vehicle labels, where the model predicted 6 instances as "walk" but the true label was "private vehicle". Below the tree structure of the pruned tree is illustrated.

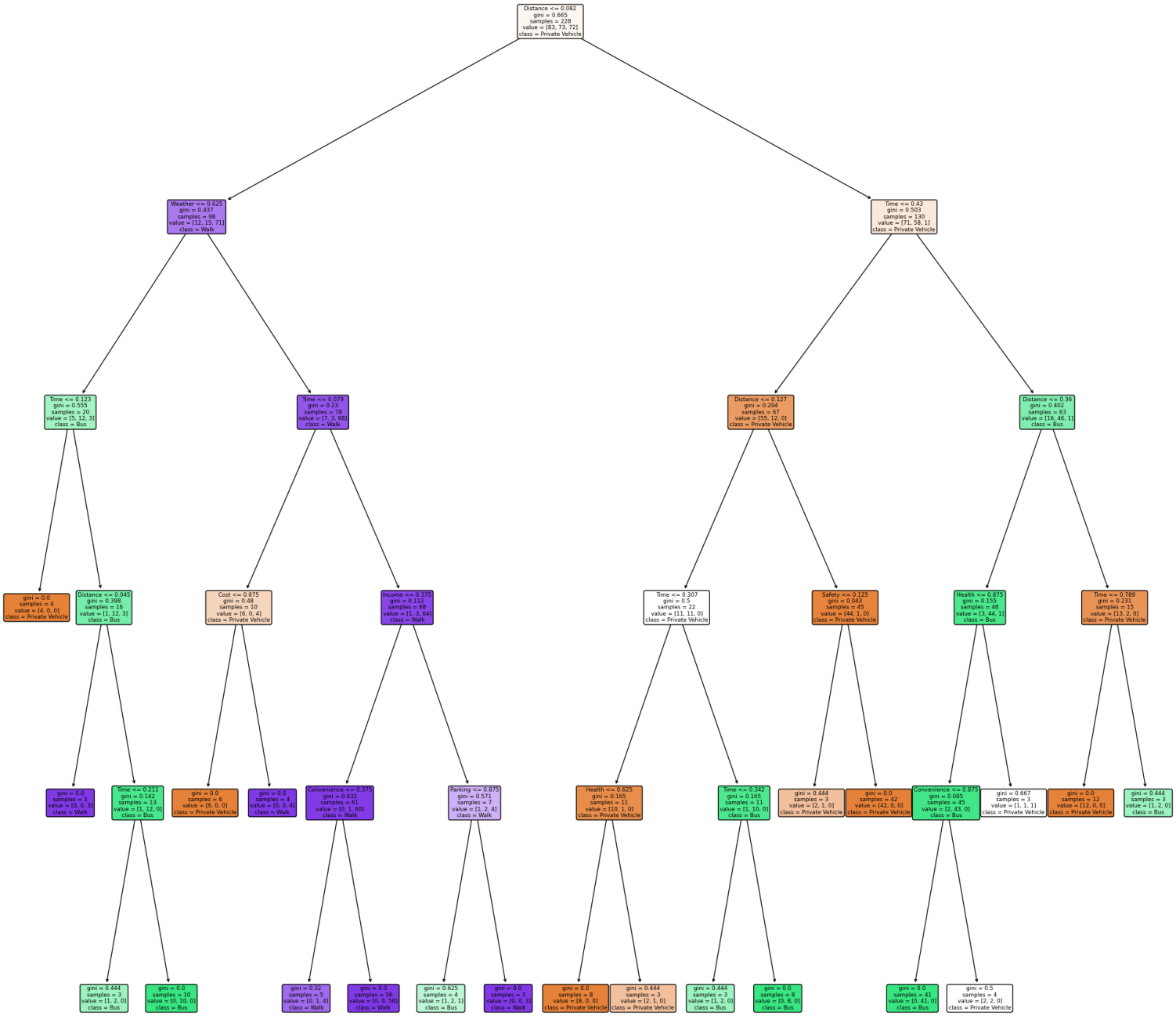


Figure 46. Tuned Tree structure

Finally, the feature importances of the model were retrieved using the feature_importances_ function. Results are illustrated below:

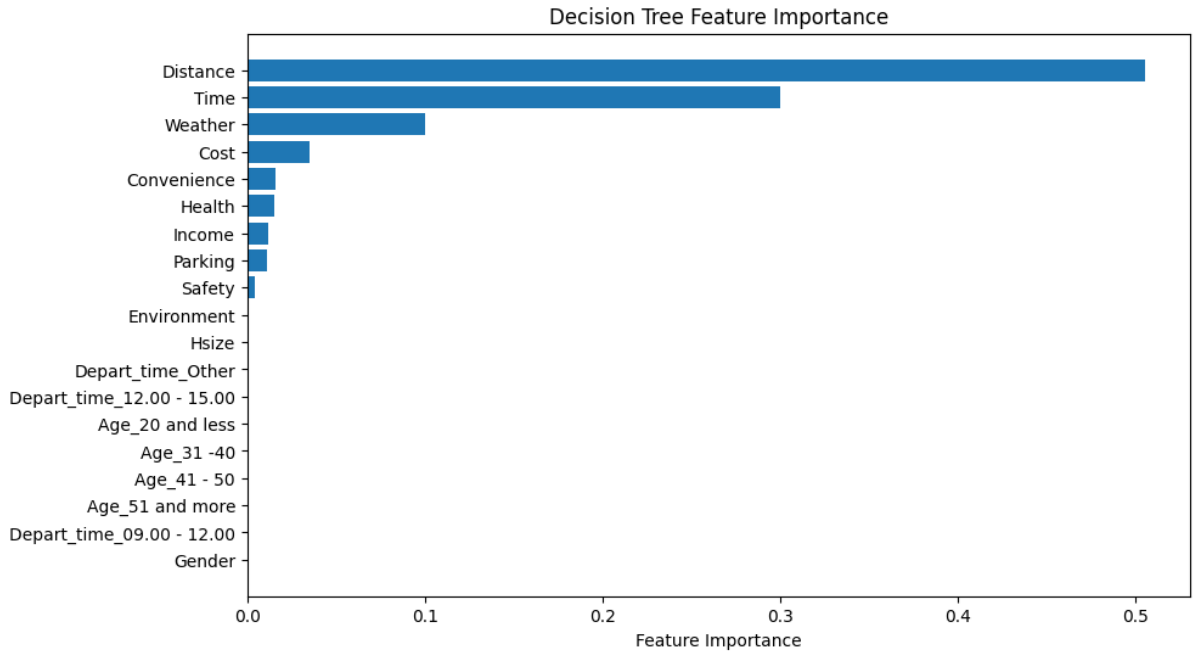


Figure 47. Decision Tree feature importance

The feature importance essentially shows how much each feature contributed to the splits that the tree made. The plot above reveals that Distance and Time features are identified as the most crucial factors for the models, followed by Weather and Cost. It is noteworthy that only 9 features out of the 19 available were selected to construct the decision tree.

The next experimentation was using the Geodesic_distance instead of the Distance feature. The model was trained on the training data and evaluated on the test set. The results are illustrated below.

Table 6. Decision Tree - Geodesic Distance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.82	0.89	0.85	0.82
Bus	0.78	0.82	0.80	
Walk	0.86	0.73	0.79	
Macro Average	0.82	0.81	0.81	

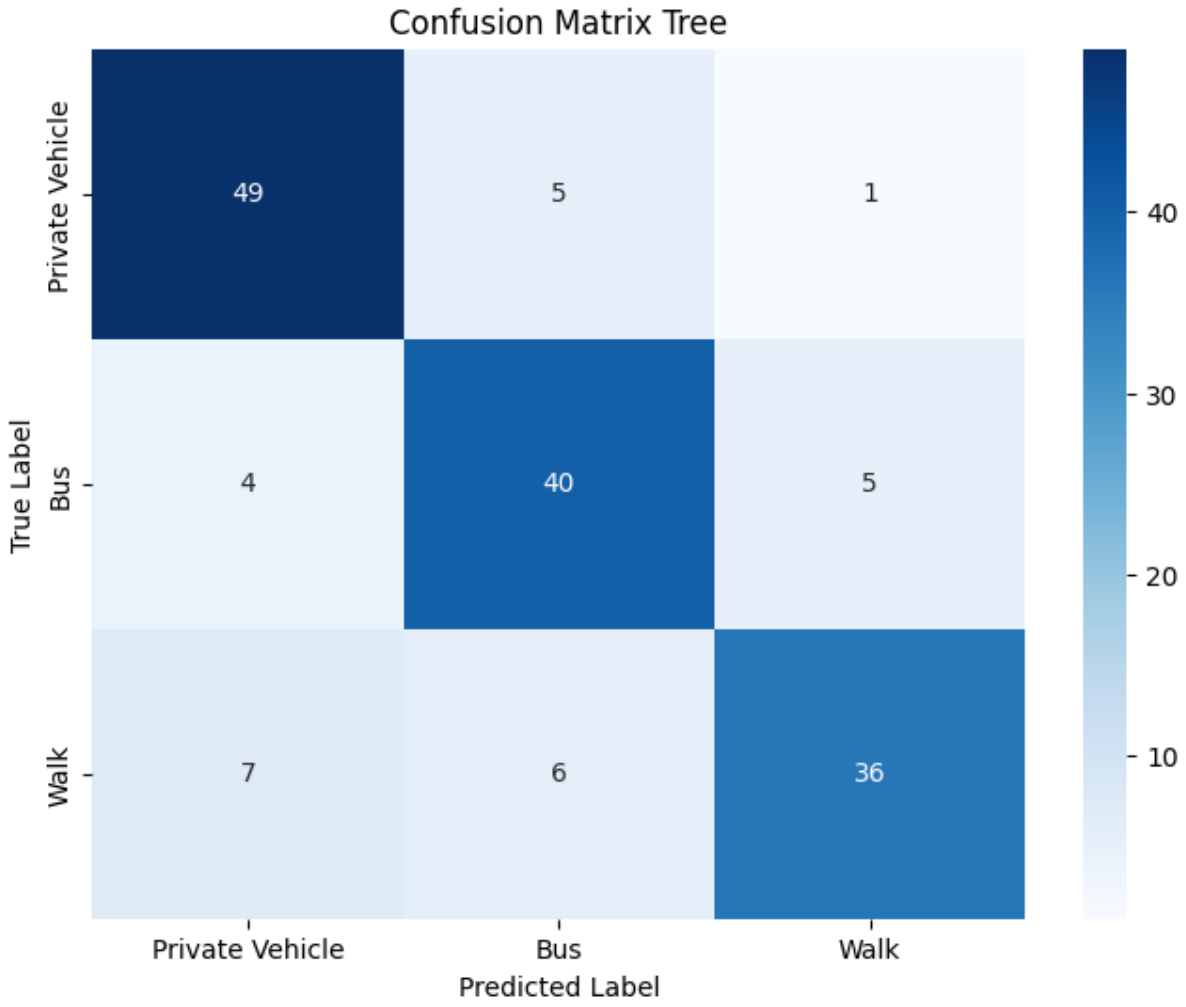


Figure 48. Confusion Matrix - Tree - Geodesic Distance

The accuracy of the model using the geodesic_distance was at 0.82. Recall for private vehicle and bus were at 0.89 and 0.82 respectively. Walk has the lowest recall for 0.73, with a trade off in precision which is the highest with 0.86. The highest misclassification is between private vehicle and walk as the model predicted 7 instances as private vehicle while the true label was walk. Again, training accuracy is at 1 so parameter tuning was explored to identify if the performance would increase. The tree was again pruned using cross validation. The parameters explored are illustrated below.

```
dt_model = DecisionTreeClassifier(random_state=21)
param_grid = {
    'max_depth': [2, 3, 4, 5, 6, 7, 8],
    'min_samples_split': [1, 2, 3, 4,  5, 6, 7, 8, 9, 10],
    'min_samples_leaf': [1, 2, 3, 4, 5, 6, 7],
}
```

The best parameters were **max depth = 6**, **min samples leaf =2** and **min samples split =2** with a **validation accuracy of 0.837**. The best model was then retrieved and evaluated on the test set. The results are depicted below.

Table 7. Pruned Tree - Geodesic Distance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.81	0.87	0.84	0.82
Bus	0.78	0.82	0.80	
Walk	0.88	0.78	0.83	
Macro Average	0.83	0.82	0.82	

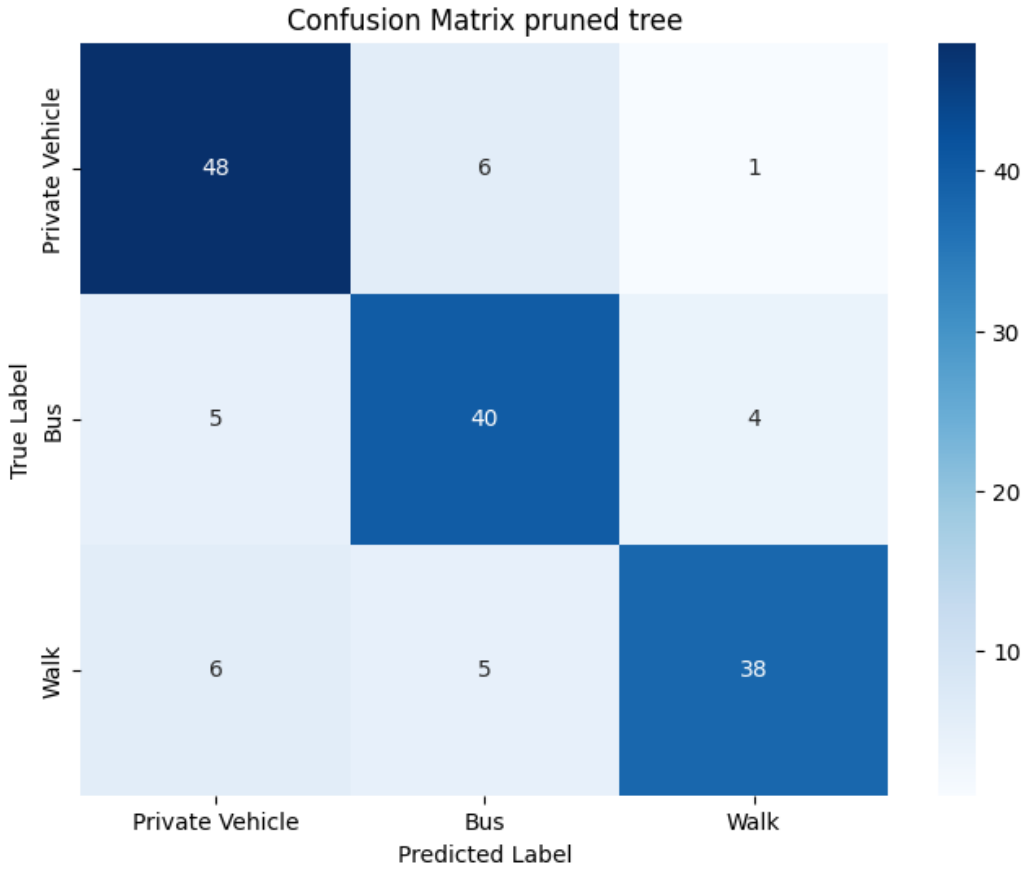


Figure 49. Confusion Matrix - Pruned Tree - Geodesic Distance

The overall accuracy of the model remained the same at 0.82 while training accuracy dropped from 1 to 0.95. Recall for walk increased from 0.73 to 0.78 while for private vehicle in dropped from 0.89 to 0.87. Precision also increased for walk from 0.86 to 0.88. Both Recall and Precision remained the same for Bus. Six instances were misclassified as private vehicle and bus while the true labels were walk and private vehicle respectively. The tree structure using Geodesic_distance is depicted below.

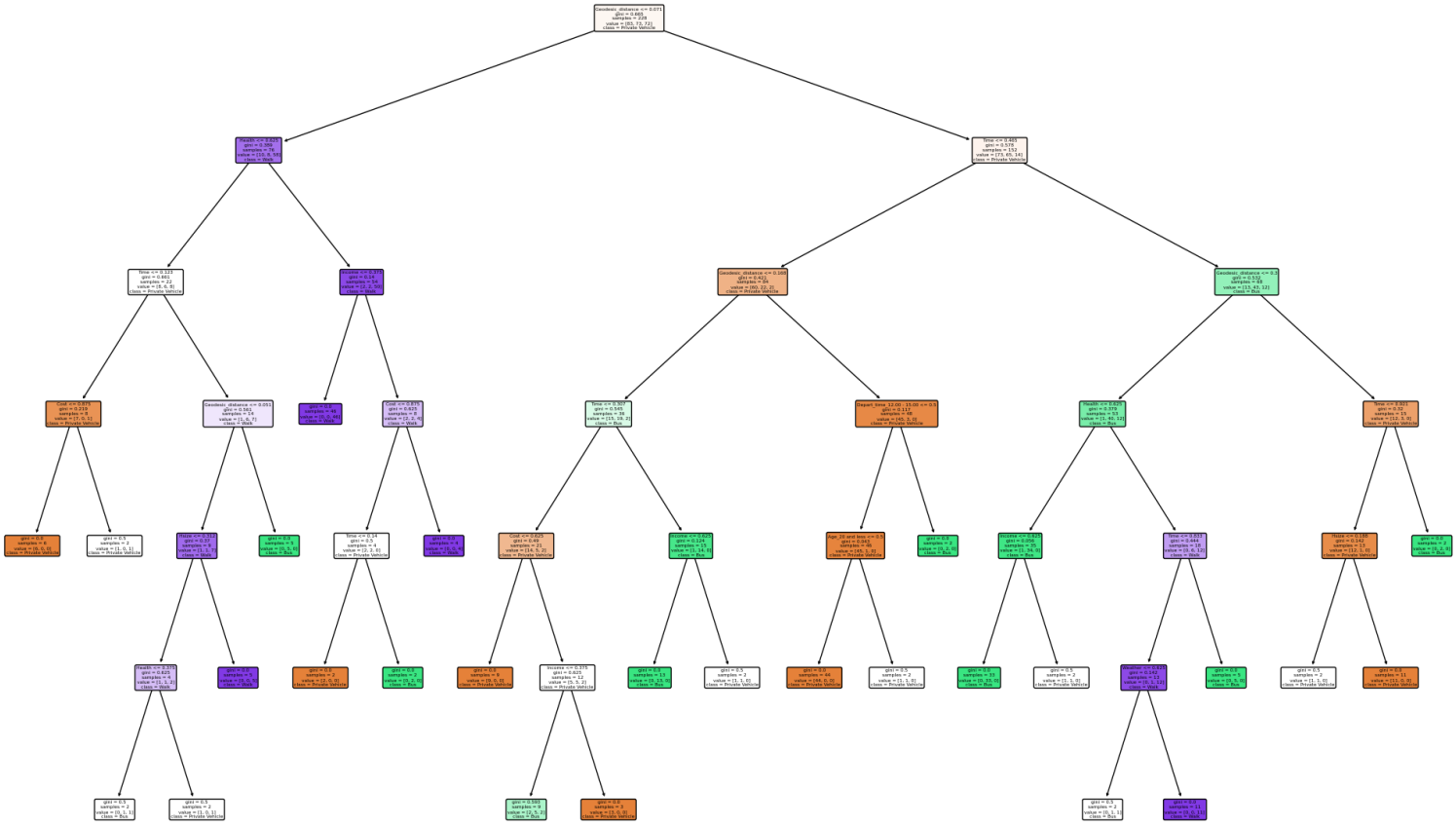


Figure 50. Tree structure - Geodesic Distance

Finally, the feature importances were extracted for the tree using the geodesic distance. The results are depicted below.

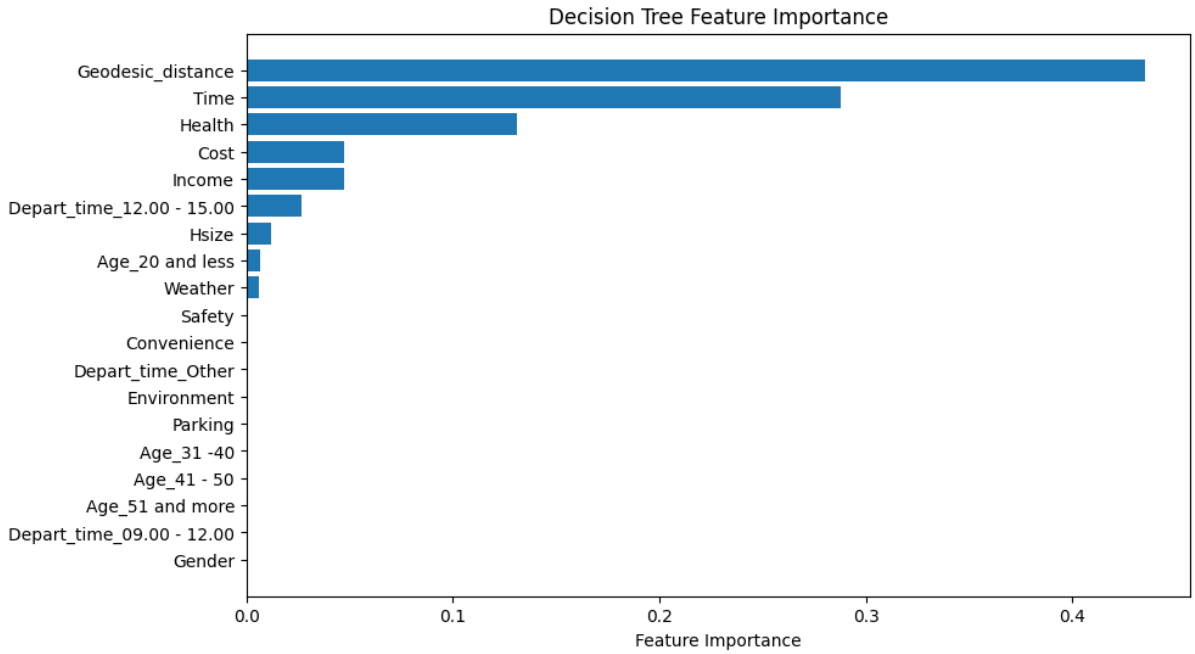


Figure 51. Tree feature importance - Geodesic Distance

Following the trend of the previous model, geodesic distance and time are the most important features for constructing the tree, followed by health cost and income. The model used 9 features in total, out of 19 for constructing the tree like the previous tree utilizing Distance instead.
The following table illustrates the metrics for the two models using different distance metrics.

Table 8. Pruned Tree evaluation (Distance vs Geodesic Distance)

	Pruned Tree (Distance)			Pruned Tree (Geodesic distance)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.82	0.89	0.80	0.81	0.78	0.88
Recall	0.82	0.80	0.88	0.87	0.82	0.78
F1-score	0.82	0.84	0.83	0.84	0.80	0.83
Accuracy	0.83			0.82		
AUC-Macro	0.90			0.88		

Swapping geodesic distance significantly dropped recall for walk labels, from 0.88 to 0.78. In contrast, there is a slight increase for private vehicle and bus in terms of recall. The overall accuracy using geodesic distance is slightly lower 0.82 compared to 0.83. The auc macro score, representing the average area under curve score for each class, is also slightly higher using distance instead of geodesic. The first model demonstrates a more balanced performance overall.

3.2.5. Random Forest

The next model that was implemented was that of Random Forest. The RandomForestClassifier() function of sklearn was used to create the model. The model was fitted on the training data, followed by evaluation on the test set. The results are illustrated below:

Table 9. Random Forest performance - Default

	Precision	Recall	F1	Accuracy
Private Vehicle	0.87	0.95	0.90	0.88
Bus	0.95	0.78	0.85	
Walk	0.85	0.92	0.88	
Macro Average	0.89	0.88	0.88	

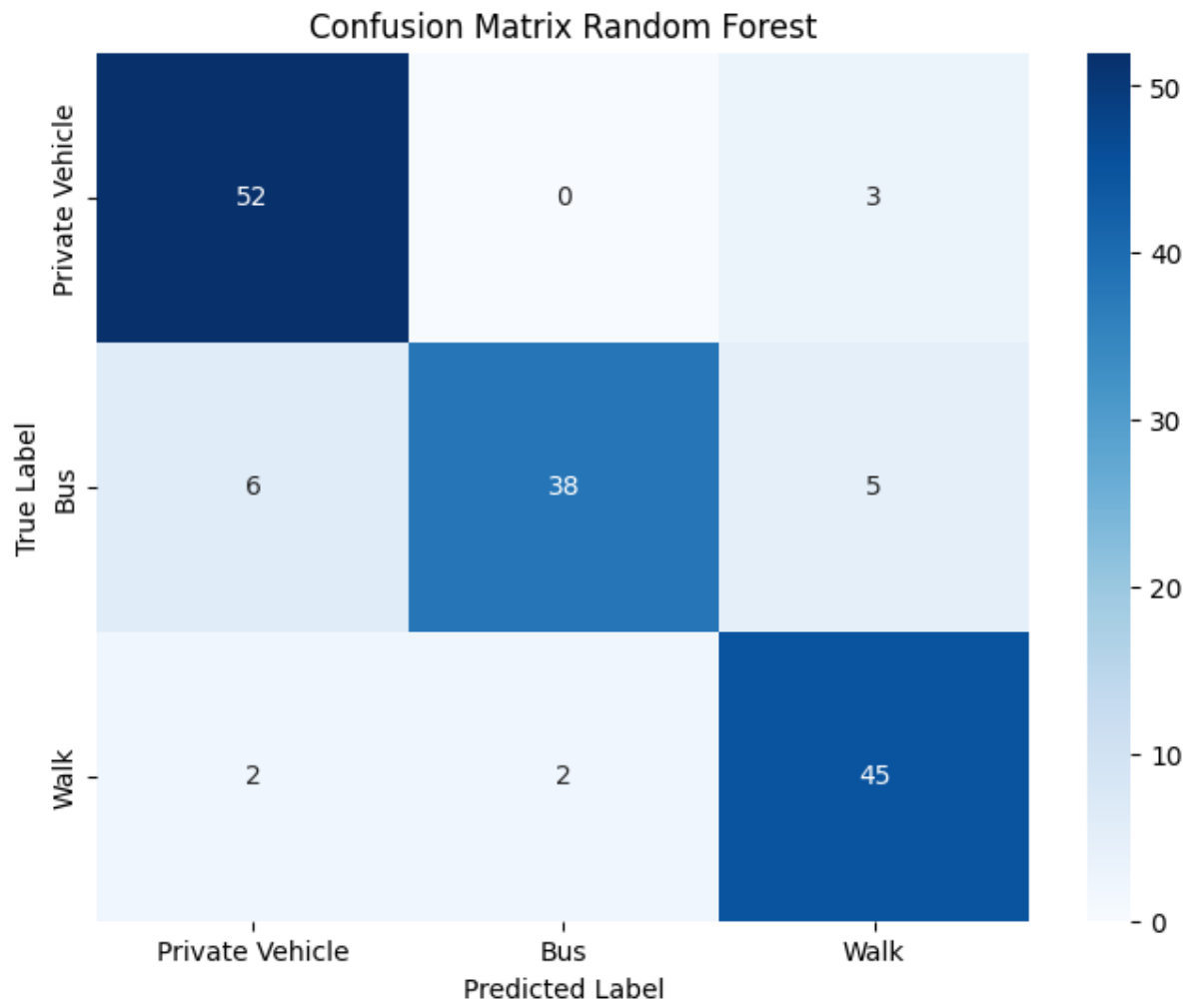


Figure 52. Confusion Matrix - Random Forest - Default

The out of bag error was 0.1053. The overall accuracy of the basic model is 0.88, correctly predicting 135 out of 153 total instances. Additionally, the model accurately predicted 38 out of 49 (0.78) cases as "bus" when the true label was "bus." Recall for private vehicle and walk was 0.95 and 0.92 respectively. The highest precision is for bus mode accurately predicting 38 out of 40 predictions made as bus. The highest misclassification is between bus and private vehicle labels as the model predicted 6 instances as private vehicle while the true label was bus. The default values of the model are max_features = "auto" and n_estimators = 100 and max_depth='None'. The max_features parameter refers to the features considered each time a split occurs on the decision tree, while "auto" is the square root of the total number of features. In contrast, the n_estimators refer to the number of trees built. Max_depth, which is the depth of each individual tree, is set to None indicating that each individual tree grows without limit. Similar to the decision tree, the GridSearchCV() function was used to tune the model and find the optimal parameters attempting to increase the performance of the model.

The process was run for each parameter individually to identify optimal area for a final grid search. Since the train set contains 19 total features, max features parameter was set at a range between 2 and 19. The validation results are depicted below.

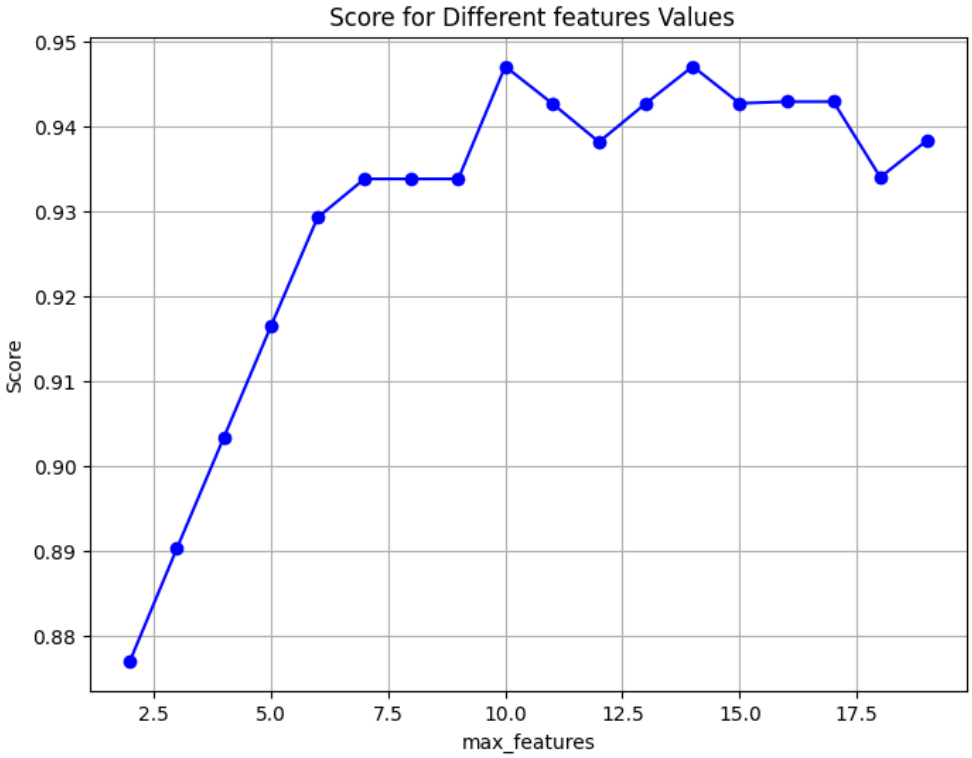


Figure 53. Grid Search RF - Max features

Looking at the plot above, it is evident that the validation accuracy is higher when the number of features is above 6, with the best overall performance to be for 10 number of features and accuracy above 0.94. The next parameter tuned was the number of estimators. The range was set between 50 and 300 number of trees. The validation results are illustrated below.

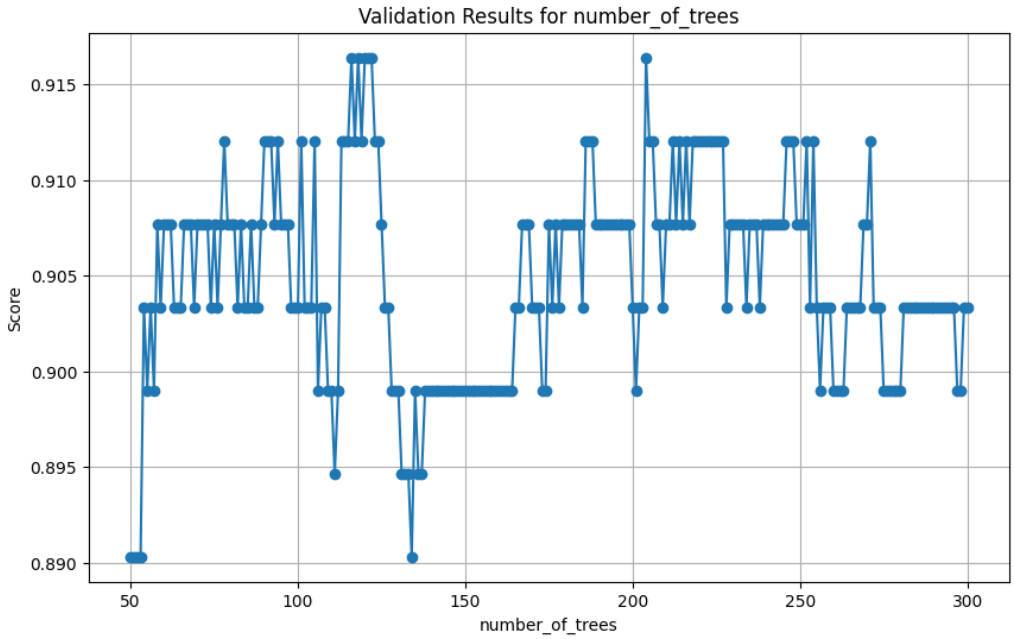


Figure 54. Grid Search RF - Number of estimators

The optimal area, where the validation accuracy is higher, is between 75 and 125 number of trees. The final parameter that was tuned was the max depth of the individual tree. The results are illustrated below.

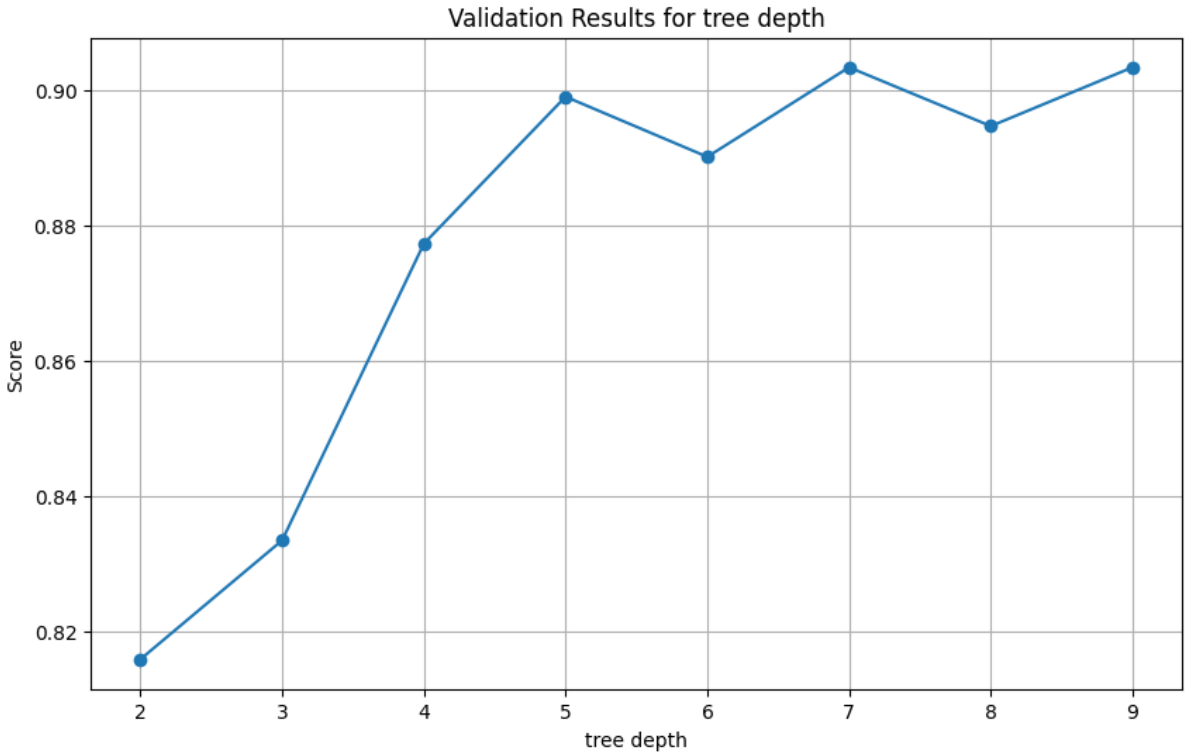


Figure 55. Grid Search RF - Tree depth

The random forest model performs best for a tree depth 7 and 9. After finding the optimal areas, a final grid search was executed with the following parameter values.

```
param_grid = {
    'n_estimators': list(range(75, 126)),
    'max_features': [10, 14],
    'max_depth' : [7, 9]}
```

The results of the final grid search were **max_features = 10** and **n_estimators = 75, and max_depth=7**. Validation accuracy was at 0.95, while the out of bag error was reduced from 0.1053 to 0.0746. The validated model was then retrieved and evaluated on the test set. The results are illustrated below.

Table 10. Tuned Random Forest performance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.96	0.93	0.94	0.92
Bus	0.95	0.86	0.90	
Walk	0.84	0.96	0.90	
Macro Average	0.92	0.91	0.91	

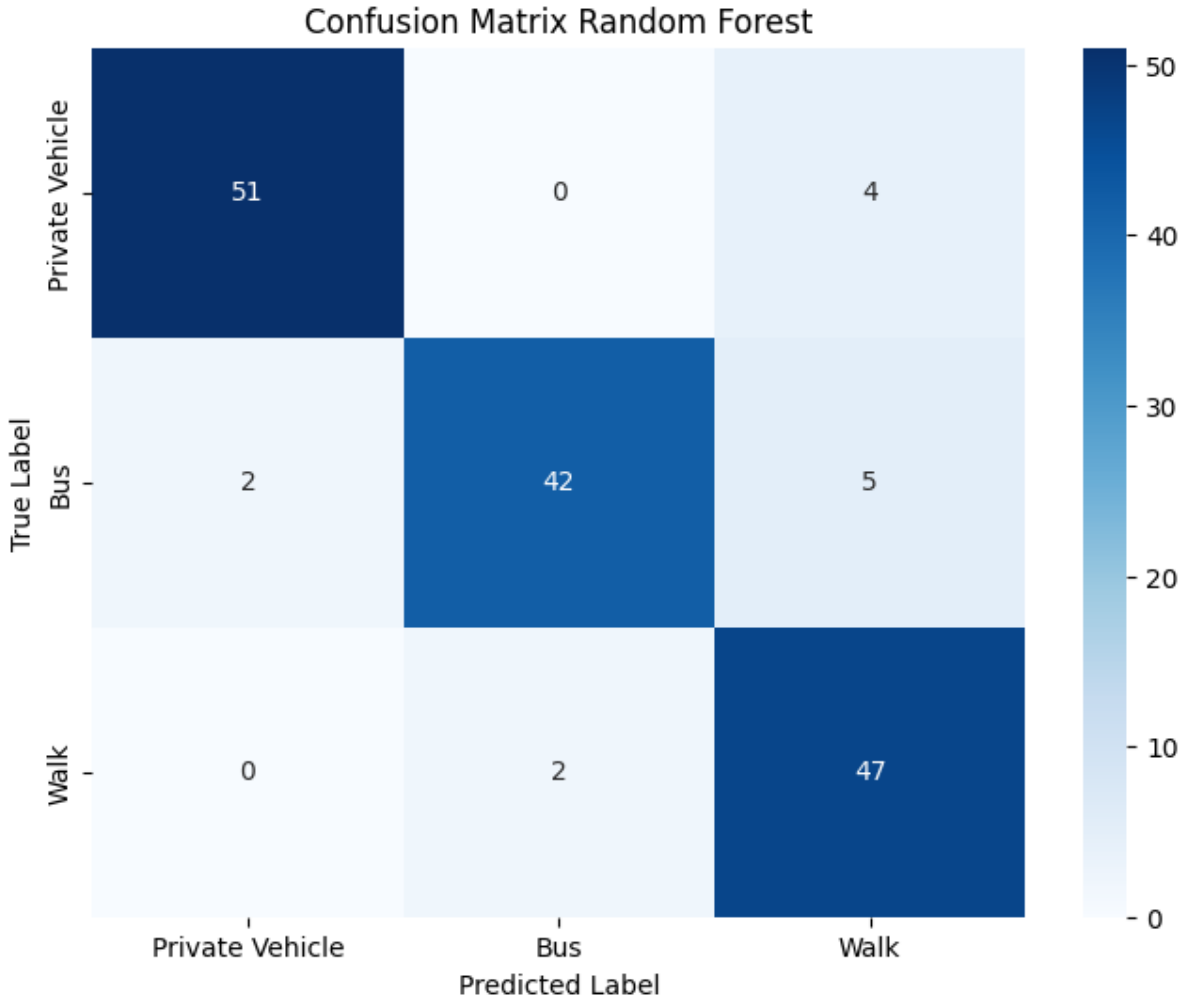


Figure 56. Confusion Matrix - Tuned Random Forest

The overall accuracy of the tuned model increased from 0.88 to 0.92. Recall for bus increased from 0.78 to 0.86 while precision remained at 0.95. Recall was also increased for walk, from 0.92 to 0.96 with a minor drop in precision from 0.85 to 0.84. Precision significantly increased for private vehicle, from 0.87 to 0.96 with a minor decrease in recall from 0.95 to 0.93. The highest misclassification occurs between walk and bus, where the model predicted 5 cases as walk when the true label was bus. The feature importances were also extracted for the model. The results are illustrated below.

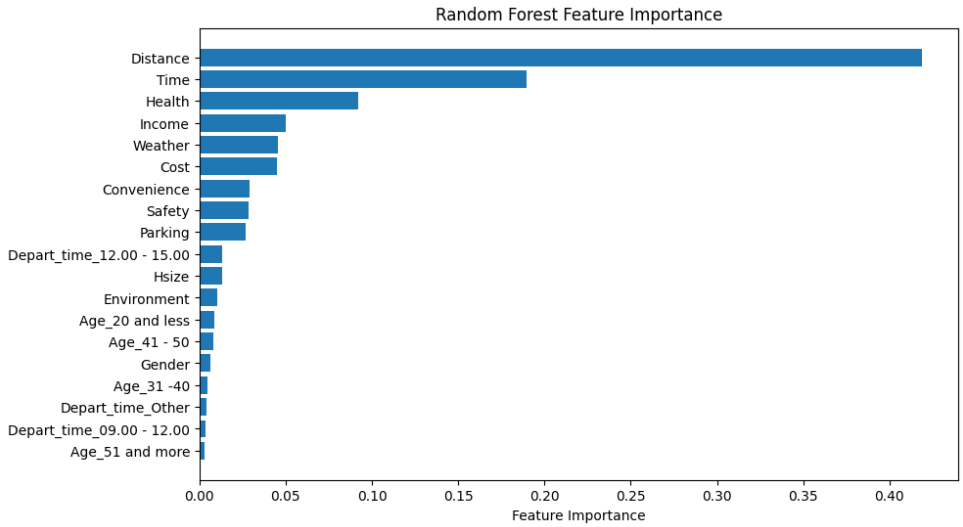


Figure 57. Random Forest feature importance

Like the single decision tree, for random forest feature importance demonstrates how much the feature contributed at the split in each of the individual tree construction. The model prioritises Distance, Time, Health, Convenience, Cost, Safety and Weather as the most crucial features. Unlike the simple tree model, the random forest has essentially utilised all parameters for split criteria at least once.

The last experimentation was swapping Distance with the Geodesic Distance feature. A new model was built, fitted on the train set, and consequently evaluated on the test set. The results are depicted below.

Table 11. Random Forest - Geodesic Distance - Default

	Precision	Recall	F1	Accuracy
Private Vehicle	0.83	0.95	0.88	0.84
Bus	0.88	0.78	0.83	
Walk	0.83	0.80	0.81	
Macro Average	0.85	0.84	0.84	

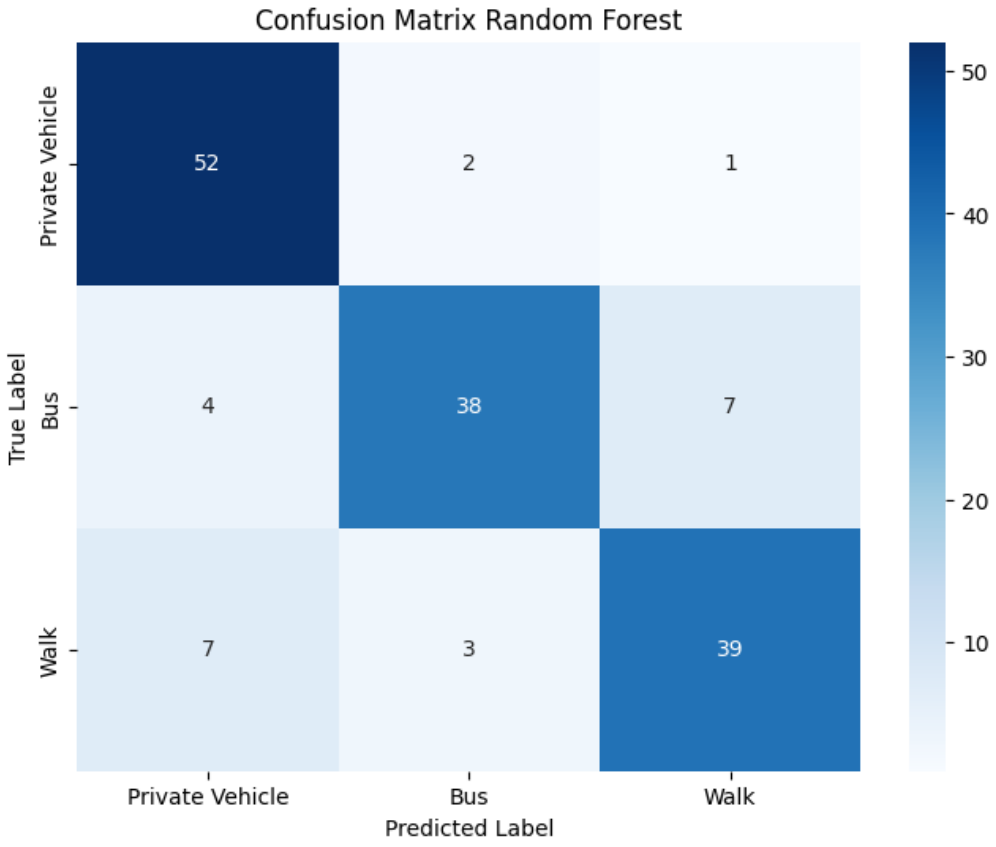


Figure 58. Confusion Matrix - Random forest - Geodesic Distance

The overall accuracy of the model was 0.84, correctly predicting 129 out of 153 instances. Recall for private vehicle was 0.95, retrieving 52 out of 55 relevant instances. Recalls for bus and walk modes were 0.78 and 0.80, retrieving 38 and 39 relevant instances respectively. The highest precision is for bus with 0.88. Swapping distances decreased the overall performance of the model, 0.84 compared to 0.88 overall accuracy of the previous default model. The next step was to tune the parameters of the model attempting to increase the performance. Again, max_depth, number of estimators and max features were configured via grid search. The results are illustrated below.

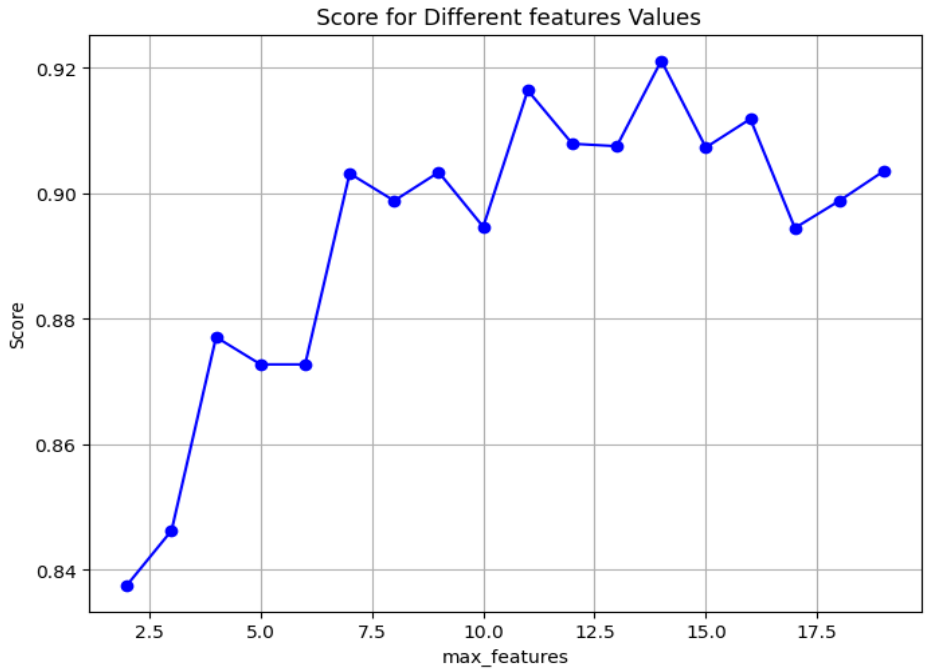


Figure 59. Grid Search RF - Max features - Geodesic Distance

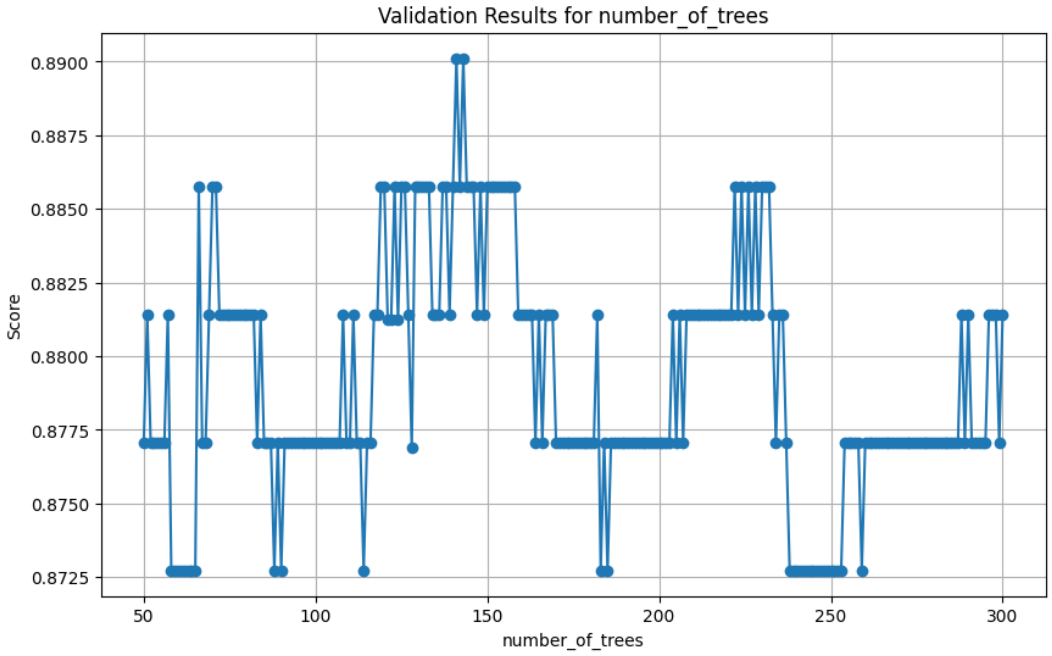


Figure 60. Grid Search RF - Number of Trees - Geodesic Distance

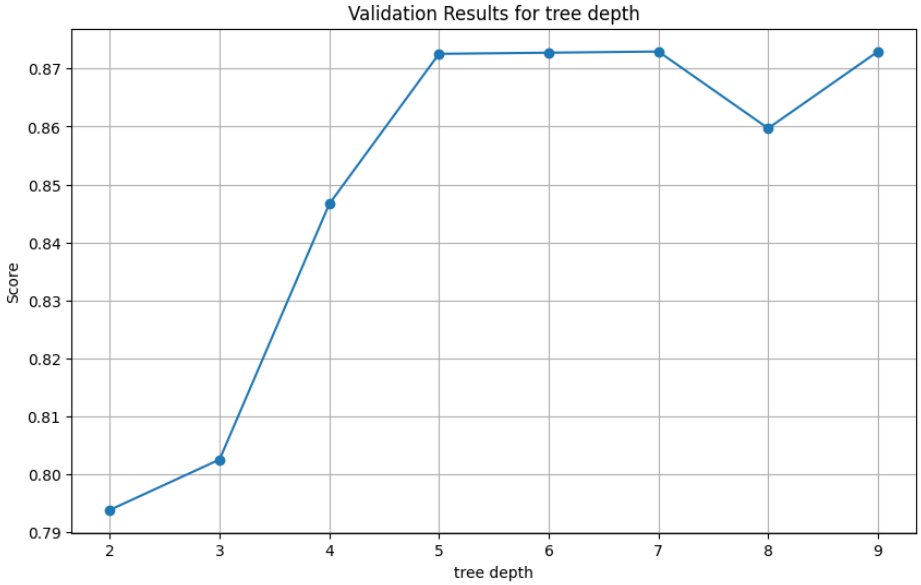


Figure 61. Grid Search RF- Max Depth - Geodesic Distance

Afte finding the optimal areas, a final grid search was configured using the following parameter values.

```
'n_estimators': list(range(125, 142)),
'max_features': [14],
'max_depth': [5, 7]}
```

The best new parameters were found for **max_depth = 5, n_estimators = 141 and max_features = 14**. The out of bag error was at 0.1272 while the **validation accuracy was at 0.90**. The tuned model was then retrieved and evaluated on the test set. The results are depicted below.

Table 12. Tuned Random Forest performance - Geodesic Distance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.88	0.95	0.91	0.86
Bus	0.88	0.78	0.83	
Walk	0.80	0.84	0.82	
Macro Average	0.86	0.85	0.85	

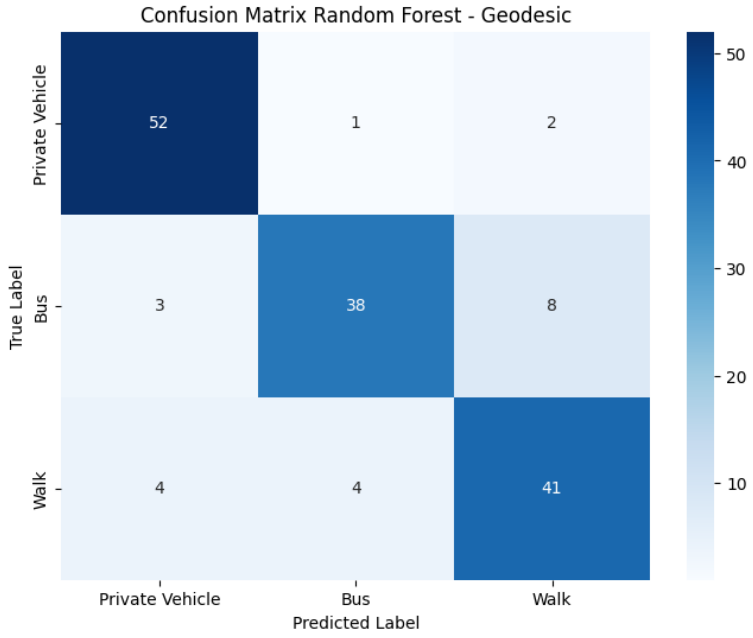


Figure 62. Confusion Matrix - Runed RF - Geodesic Distance

The overall accuracy of the tuned model increased from 0.84 to 0.86. Precision increased for private vehicle from 0.83 to 0.88 while recall remained at 0.95. Recall also increased for walk, from 0.80 to 0.84 while precision dropped from 0.83 to 0.80. All the metrics for bus remained the same. The highest misclassification occurs between walk and bus where the model predicted 8 instances as walk while the true label was bus. The feature importances for the model construction were also extracted. The results are illustrated in the plot below. The trend is likewise the previous Random Forest model, with geodesic distance at the top followed by time and Health, Cost and Income and Weather. The only feature not used in any of the individual trees was whether the respondent was 51 and more or not.

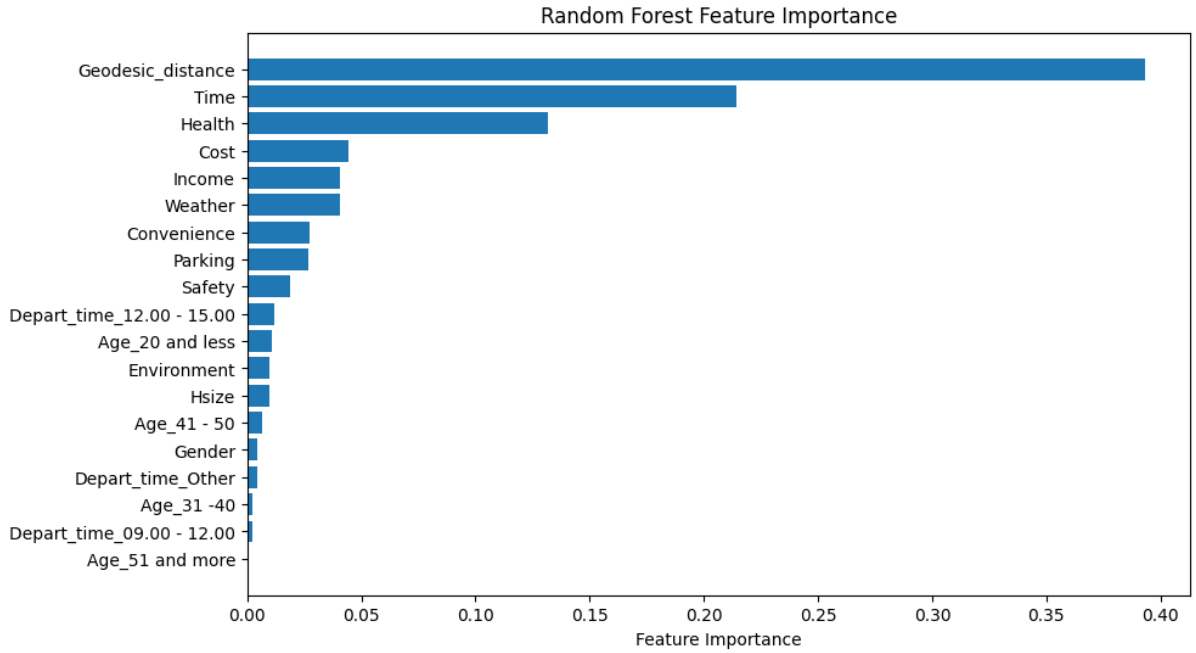


Figure 63. Random Forest feature importance - Geodesic Distance

Table 13. Random Forest (Distance vs Geodesic Distance)

	Random Forest (Distance)			Random Forest (Geodesic distance)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.96	0.95	0.84	0.88	0.88	0.80
Recall	0.93	0.86	0.96	0.95	0.78	0.84
F1-score	0.94	0.96	0.90	0.91	0.83	0.82
Accuracy	0.92			0.86		
AUC-Macro	0.983			0.96		

The Random Forest model, utilizing the Distance metric, demonstrates better performance in almost every metric. The only exception occurs for private vehicle class, aa the model utilizing geodesic distance demonstrated a recall of 0.95 compared to a recall of 0.93 of the other model. Overall, the first model has a better performance overall, compared to geodesic distance-based model.

3.2.6. XGBoost

The next model applied was that of XGboost. The model was created using the **xgb.XGBClassifier(objective='multi:softmax', num_class=3, random_state=42)** function.

The objective=”multi:softmax” parameter indicates that the task of the model is to classify more than two classes, while the num_class = 3 parameter indicates the number of classes. Similarly to random forest, the random_state parameter is set to achieve reproducibility of the results. The model was then fitted on the training set and consequently evaluated on the test set. The results are depicted below.

Table 14. XGBoost performance - Default

	Precision	Recall	F1	Accuracy
Private Vehicle	0.89	0.98	0.93	0.92
Bus	0.95	0.84	0.89	
Walk	0.92	0.92	0.92	
Macro Average	0.92	0.91	0.91	

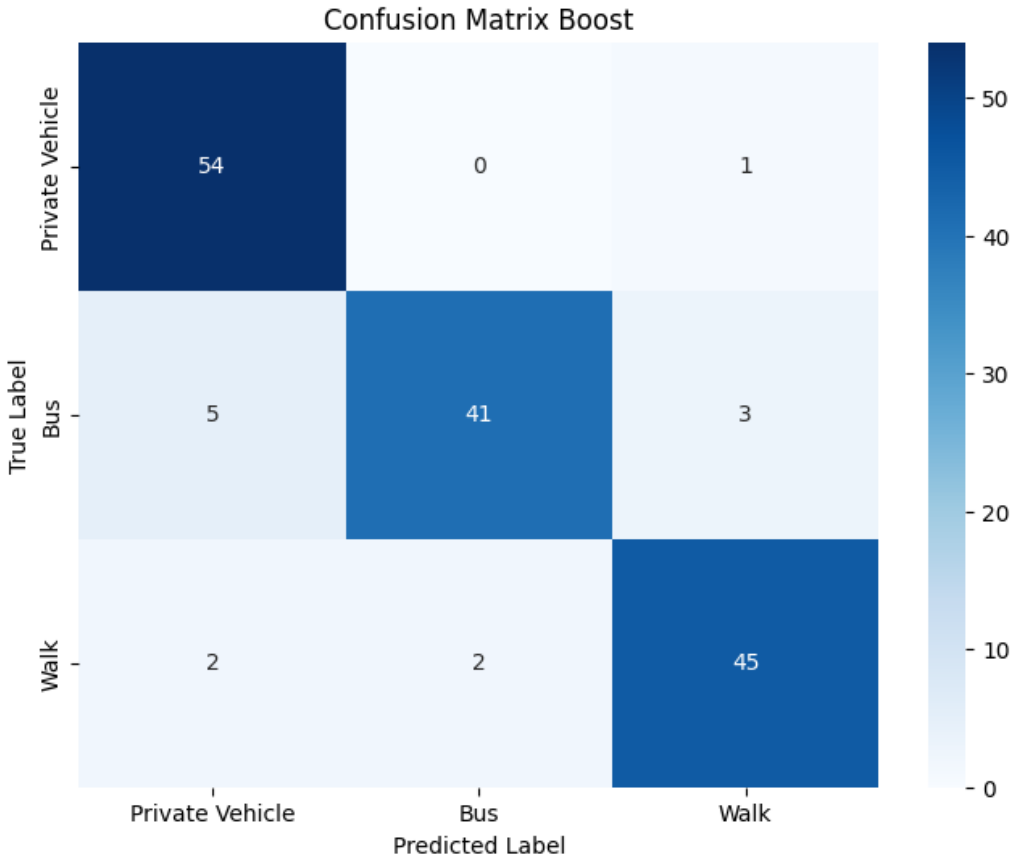


Figure 64. Confusion Matrix - XGBoost - Default

The overall accuracy of the model was 0.92, correctly predicting 140 out of 153 total instances. Recall for private vehicle and walk, was found 0.98 and 0.92 respectively. Bus has the highest precision with 0.95 but a lower recall with 0.84. The highest misclassification occurs between “bus” and “private vehicle”, as the model predicted 5 instances as “private vehicle” while the true label was “bus”. Although the model's performance appears satisfactory, various sets of parameters were modified to assess if performance could be enhanced. The default values of the model are the following: **n_estimators = 100, subsample = 1, colsample_bytree =1 , learning_rate = 0.3**. The number of estimators refers to the number of trees built like Random Forest. The subsample parameter refers to the fraction of the training data that are sampled during the training phase. The colsample_bytree parameter refers to the fraction of features that are sampled for each tree. Finally, the learning_rate parameter refers how quickly or slowly the model adapts to the patterns in the training data. The max_depth parameter was also tuned for the model. Like random forest and decision tree, tree depth controls the individual tree size. Those parameters were configured through the GridSearchCV() function individually to create a final grid and evaluate the model.

Initially, a grid search was conducted to test the number of estimators within the range of 50 to 300 trees. The outcomes are illustrated below.

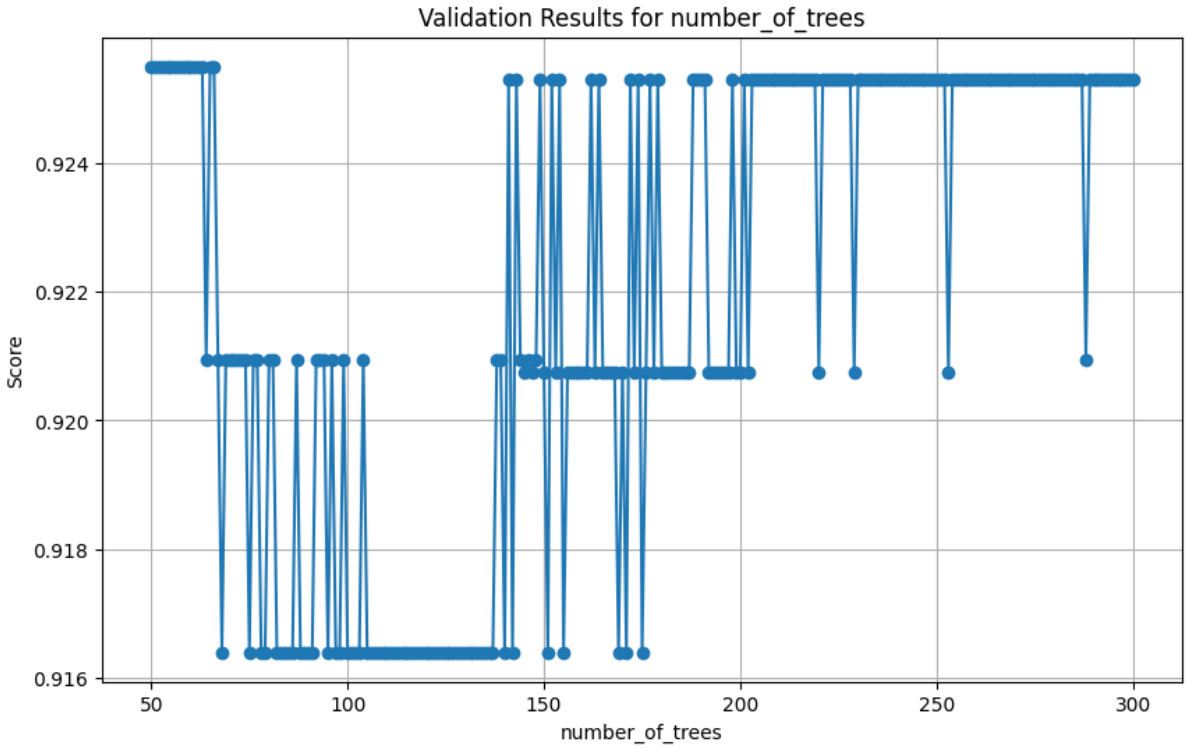


Figure 65. Grid Search XGBoost - Number of Trees

The overall validation accuracy varies between the range, though is higher between 50 and 70 number of trees. The next parameter that was tested was that of learning_rate with values from 0.1 up to 1. The results are illustrated below.

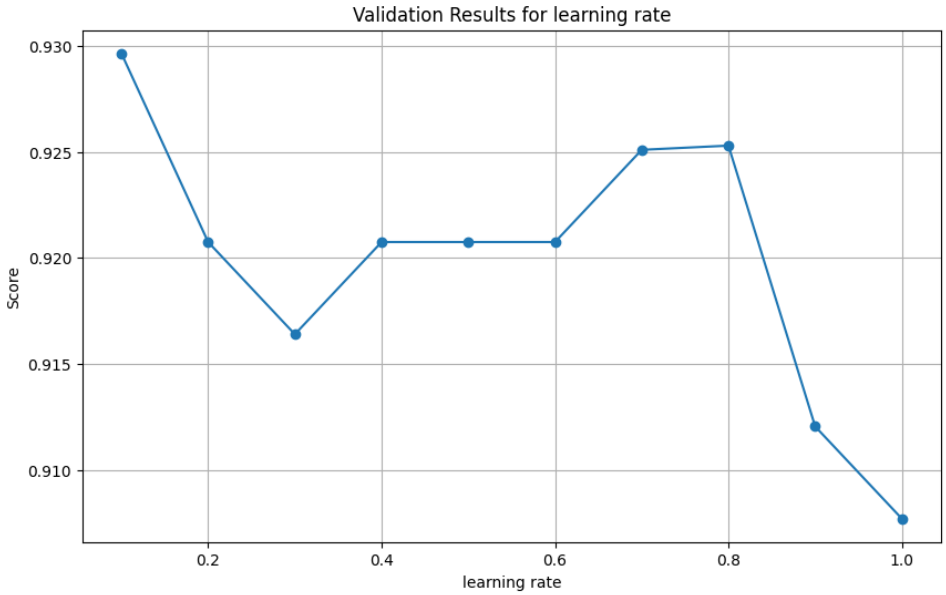


Figure 66. Grid Search XGBoost - Learning rate

The results indicate best validation accuracy for learning_rate = 0.1. The next parameter tested was subsamples, with values from 0.1 to 1. The results are illustrated below.

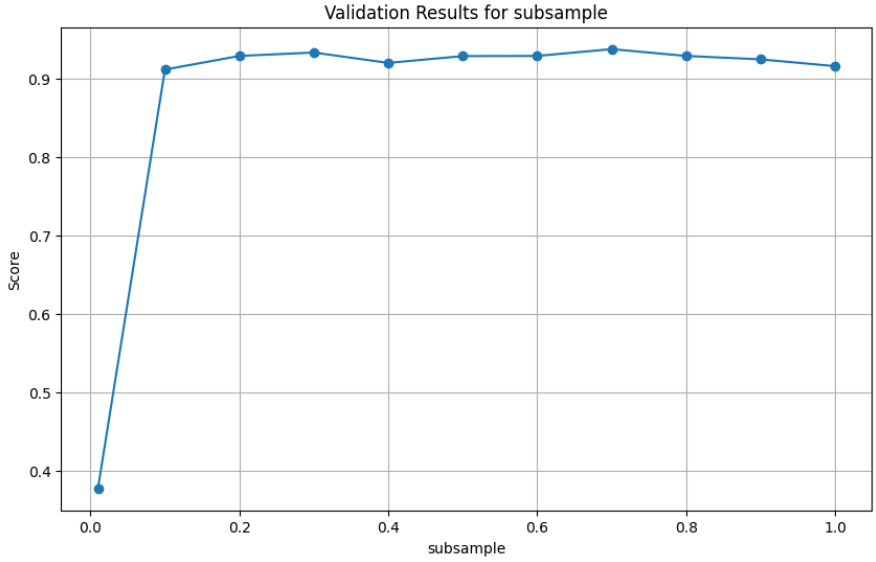


Figure 67. Grid Search XGBoost - Subsample

It is identified that the validation accuracy is best for 0.7. Though, validation accuracy is consistently high for all values of subsample. The next parameter tuned was that of colsample_bytree. The results are illustrated below.

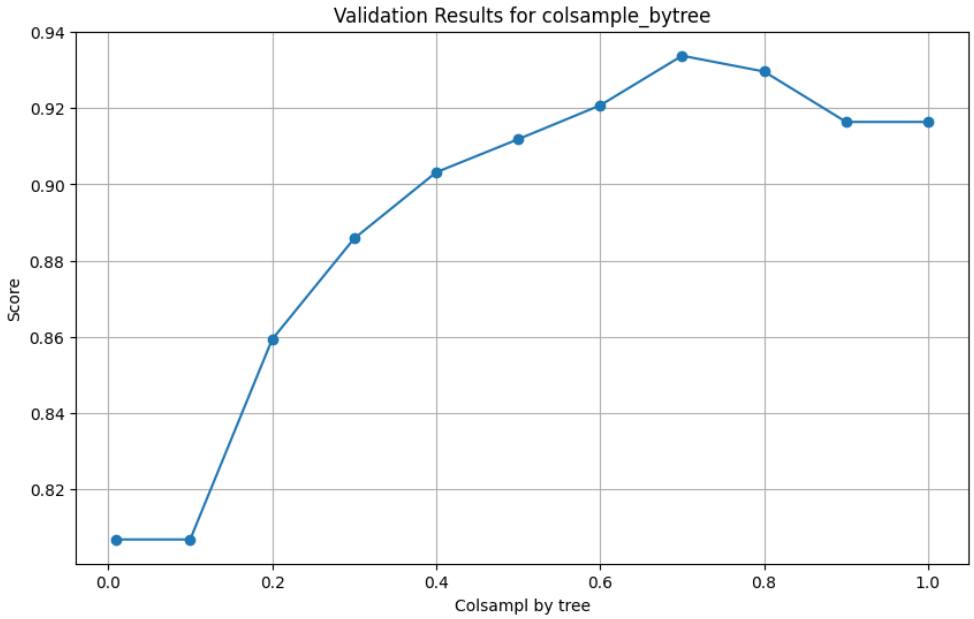


Figure 68. Grid Search XGBoost - Colsample

The validation accuracy is best for colsample value of 0.7. The last parameter tuned was that of max_depth. The results are depicted below.

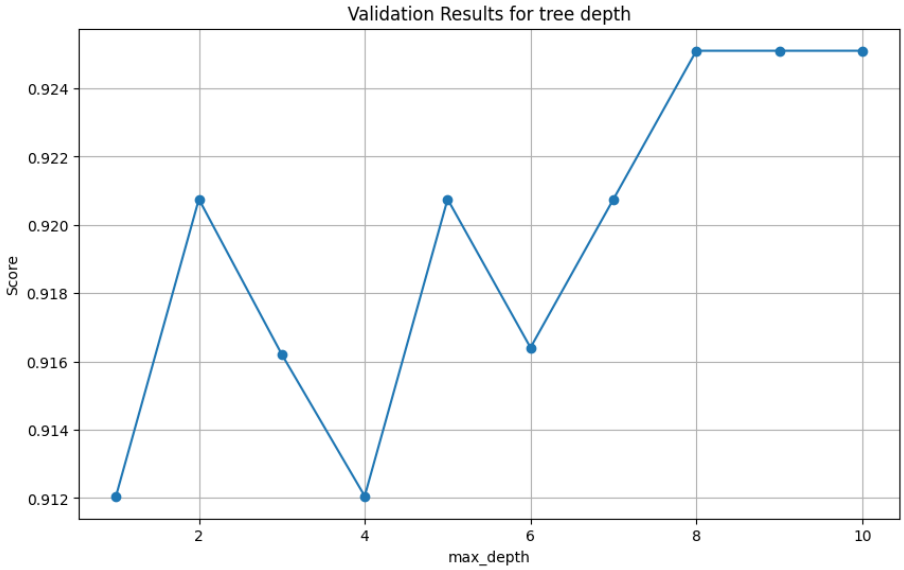


Figure 69. Grid Search XGBoost - max depth

Accuracy is higher for a max depth of 8, 9 and 10, though the parameter will be set at 8 to limit the growth of the trees.

After finding the optimal values a final grid was run to tune the model. The parameter values that were tuned through grid search are illustrated below.

```
'n_estimators': list(range(50, 101)),
'subsample': [0.7],
'colsample_bytree' : [0.7, 0.8],
'learning_rate': [0.1],
'max_depth':[8]}
```

The optimal parameters values from grid search are the following: **n_estimators = 50, subsample = 0.7, colsample_bytree = 0.8, learning_rate = 0.1, max_depth=8 and validation accuracy of 0.946**. The tuned model was then retrieved and evaluated on the test set. The results are depicted below.

Table 15. Tuned XGBoost performance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.95	0.96	0.95	0.93
Bus	0.96	0.88	0.91	
Walk	0.88	0.94	0.91	
Macro Average	0.93	0.93	0.93	

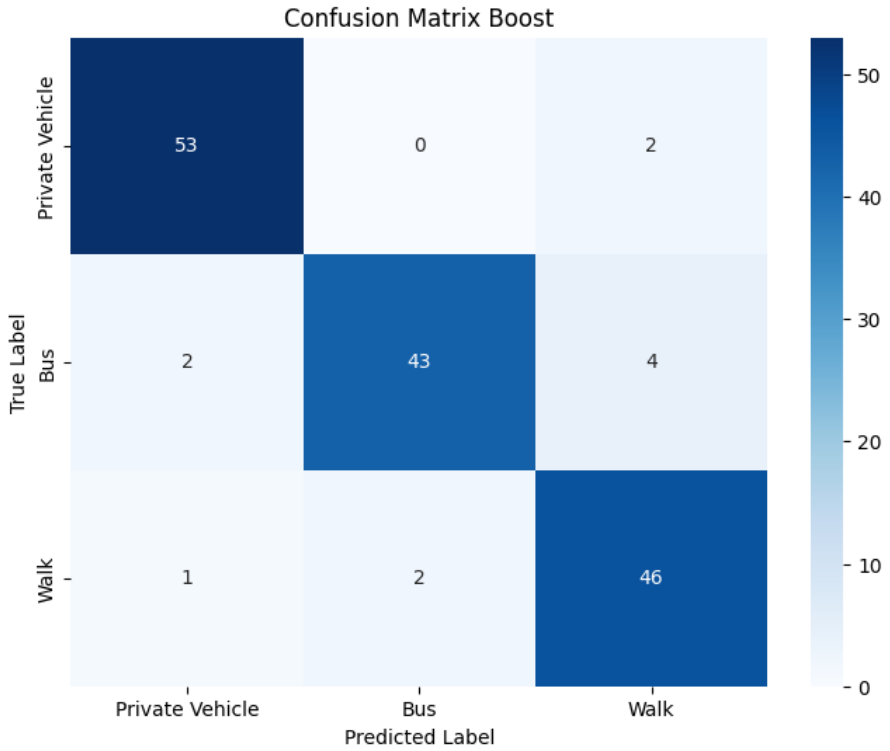


Figure 70. Confusion Matrix Tuned XGBoost

The overall accuracy of the tuned model increased from 0.92 to 0.93. Precision for private vehicle increased from 0.89 to 0.95 with a trade off in recall as it decreased from 0.98 to 0.95. Recall for bus increased from 0.84 to 0,88, while precision slightly increased from 0.95 to 0.96. Recall also increased for walk, from 0.92 to 0.94 while precision decreased from 0.92 to 0.88. The highest misclassification occurred between walk and bus, as the model predicted 4 cases as walk while the true label was bus. Overall, the tuned model demonstrates a more balanced performance compared to the default one. The feature importances were also extracted for the model. The results are depicted below.

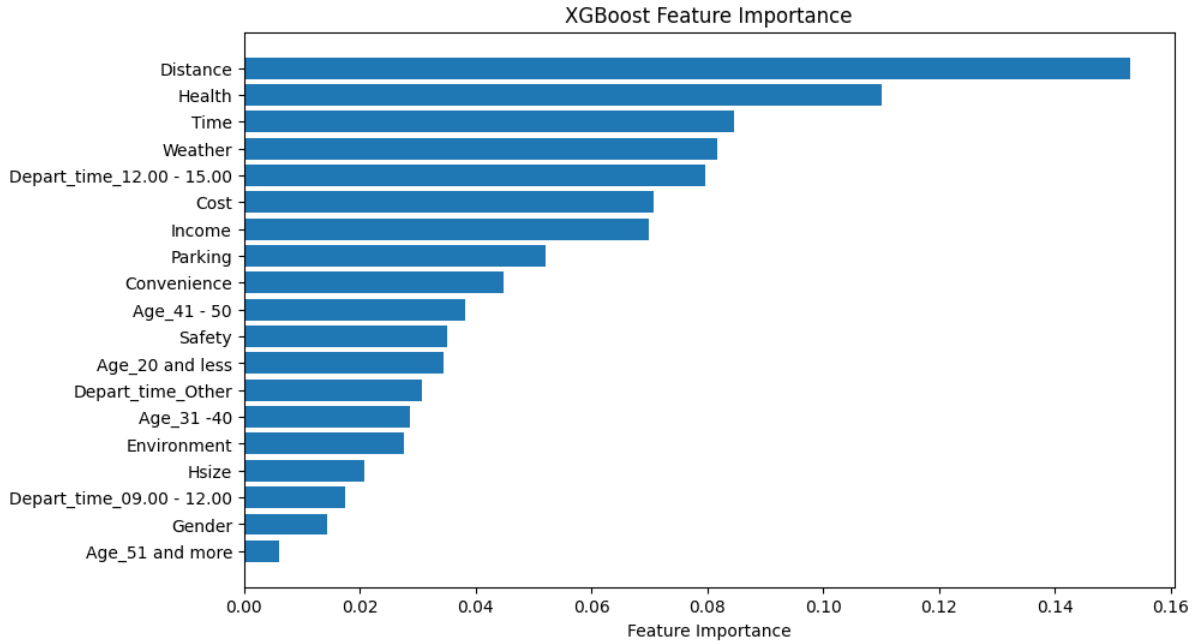


Figure 71. XGBoost feature importance

Like the previous tree-based models, distance remains the most important feature for constructing the individual trees. Surprisingly, while time was the second most important feature for all the previous models, XGboost ranks influence of physical activity and health as the second most important feature. High on the list is also one of the binary features and whether the respondent commutes to work between 12.00 – 15.00 or not. At the bottom of the list are Gender and whether the age of the commuter was more than 51 years old or not. Still, the model used all the features for the individual tree construction. The last experimentation was again swapping Distance with Geodesic_Distance. A new model was created, trained on the train set and consequently evaluated on the test set. The results are illustrated below.

Table 16. XGBoost - Geodesic Distance - Default

	Precision	Recall	F1	Accuracy
Private Vehicle	0.85	0.96	0.91	0.86
Bus	0.87	0.82	0.84	
Walk	0.87	0.80	0.83	
Macro Average	0.86	0.86	0.86	

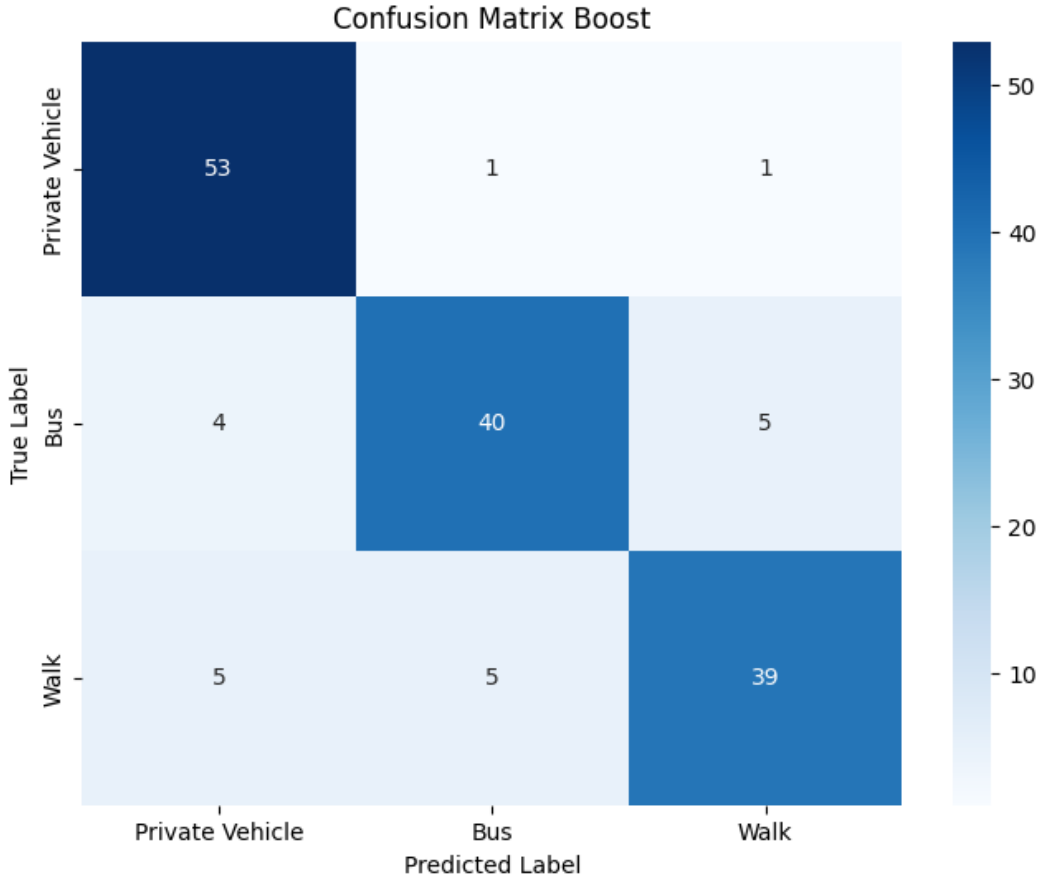


Figure 72. Confusion Matrix - XGBoost -Geodesic Distance

The overall accuracy of the model dropped from 0.92 to 0.86 by swapping distances. Recall for private vehicle was 0.96, followed by bus at 0.82 and walk with 0.80. Precision is higher for both bus and walk with 0.87. While the overall metrics of the model dropped swapping distances, the parameters were re-tuned to increase the performance of the model. The same parameters as before were tuned to find optimal areas. The validation results are illustrated below.

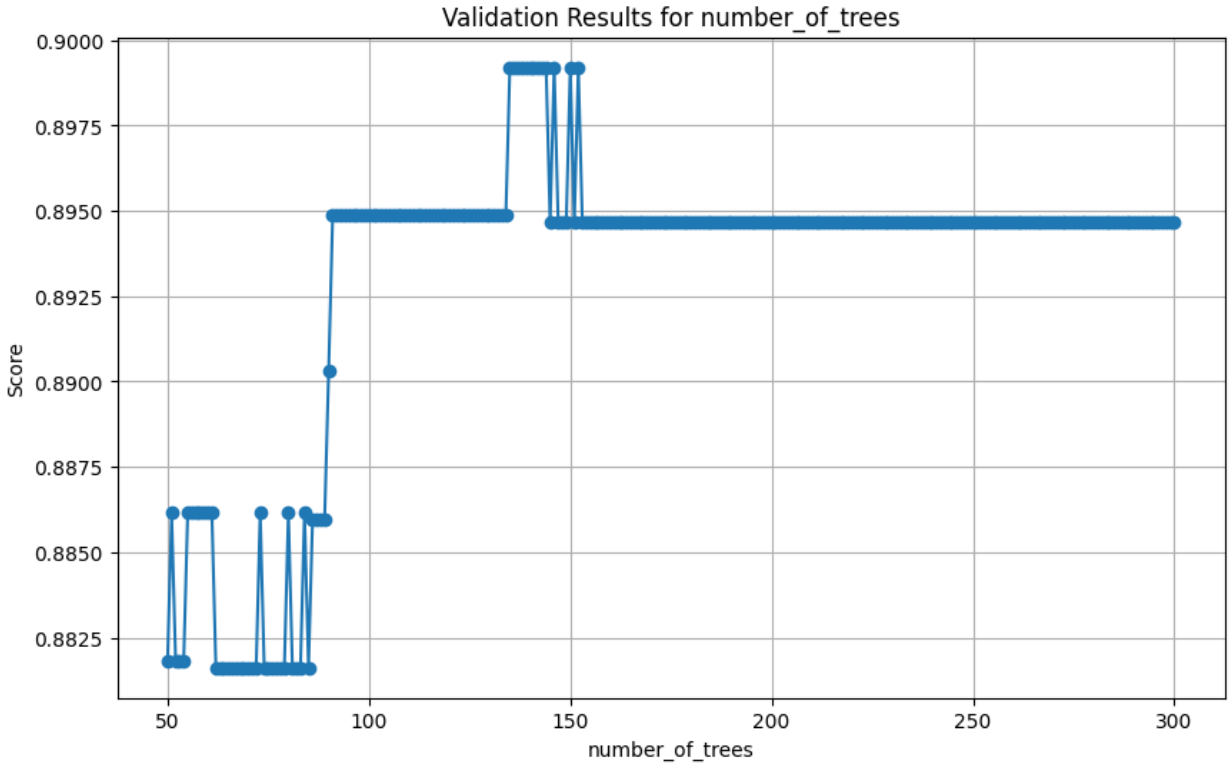


Figure 73. Grid Search XGB - Number of Trees - Geodesic Distance

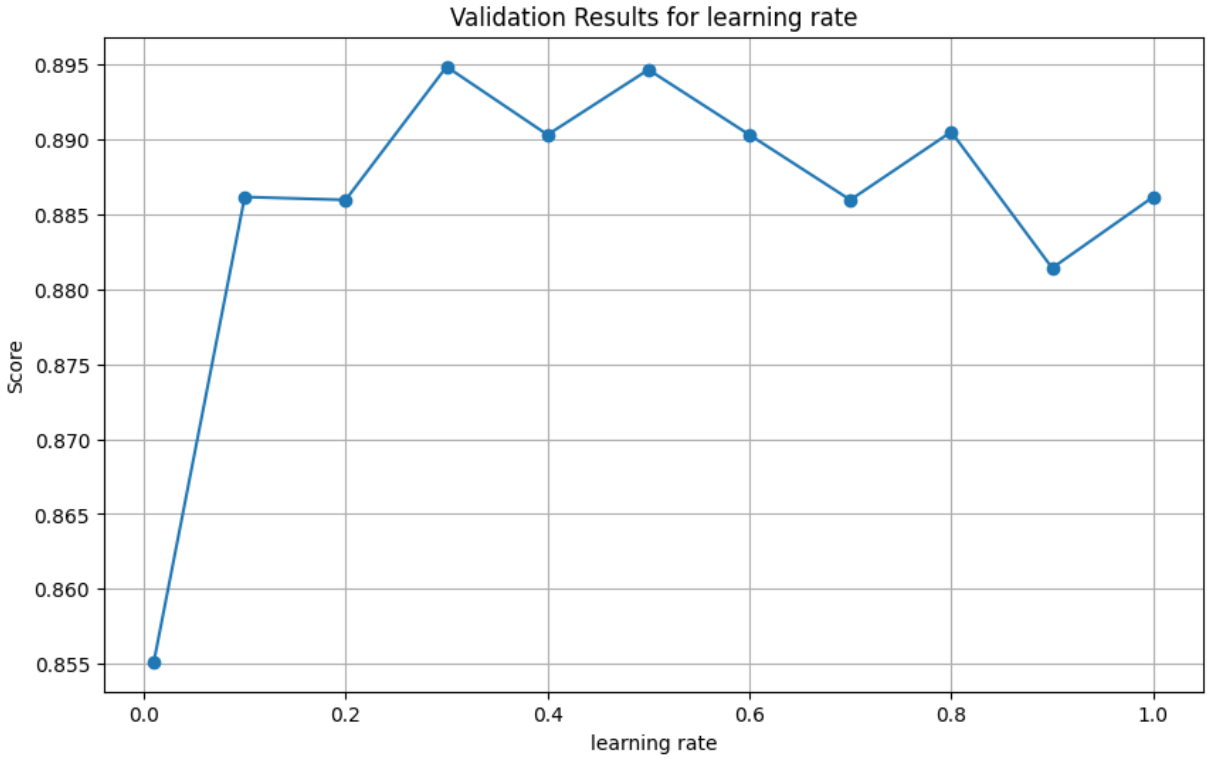


Figure 74. Grid Search XGB - Learning rate - Geodesic Distance

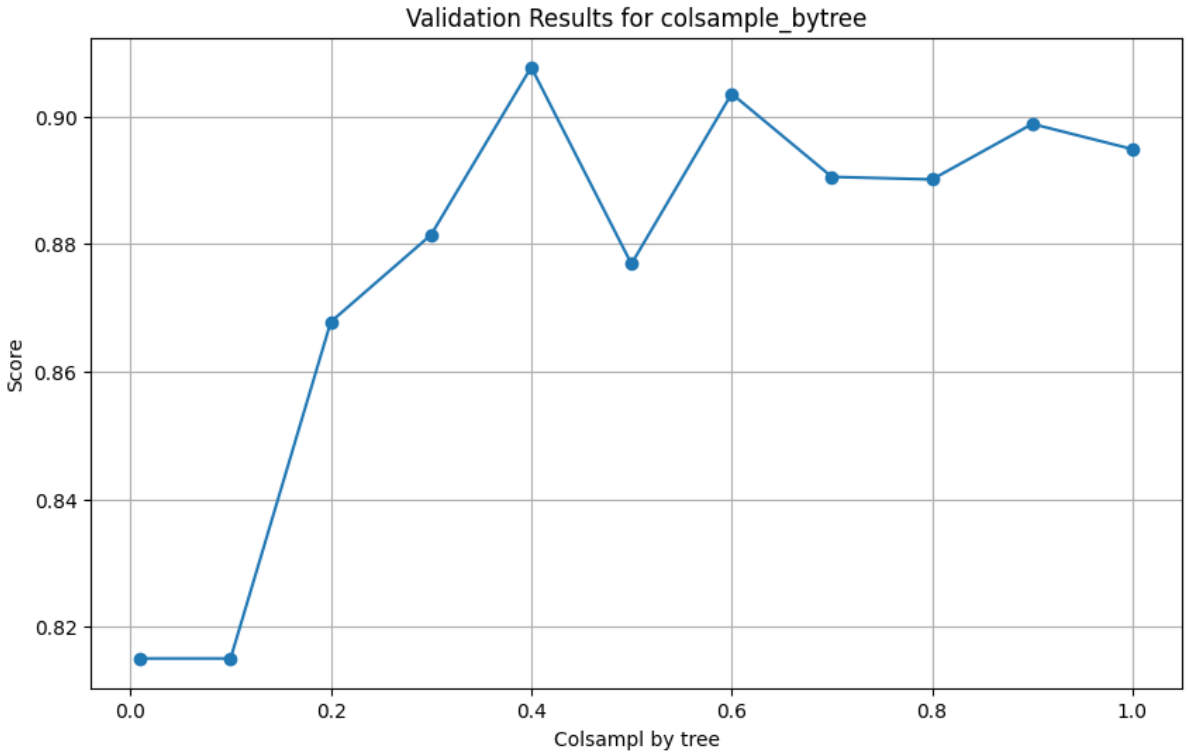


Figure 75. Grid Search XGB - Colsample - Geodesic Distance

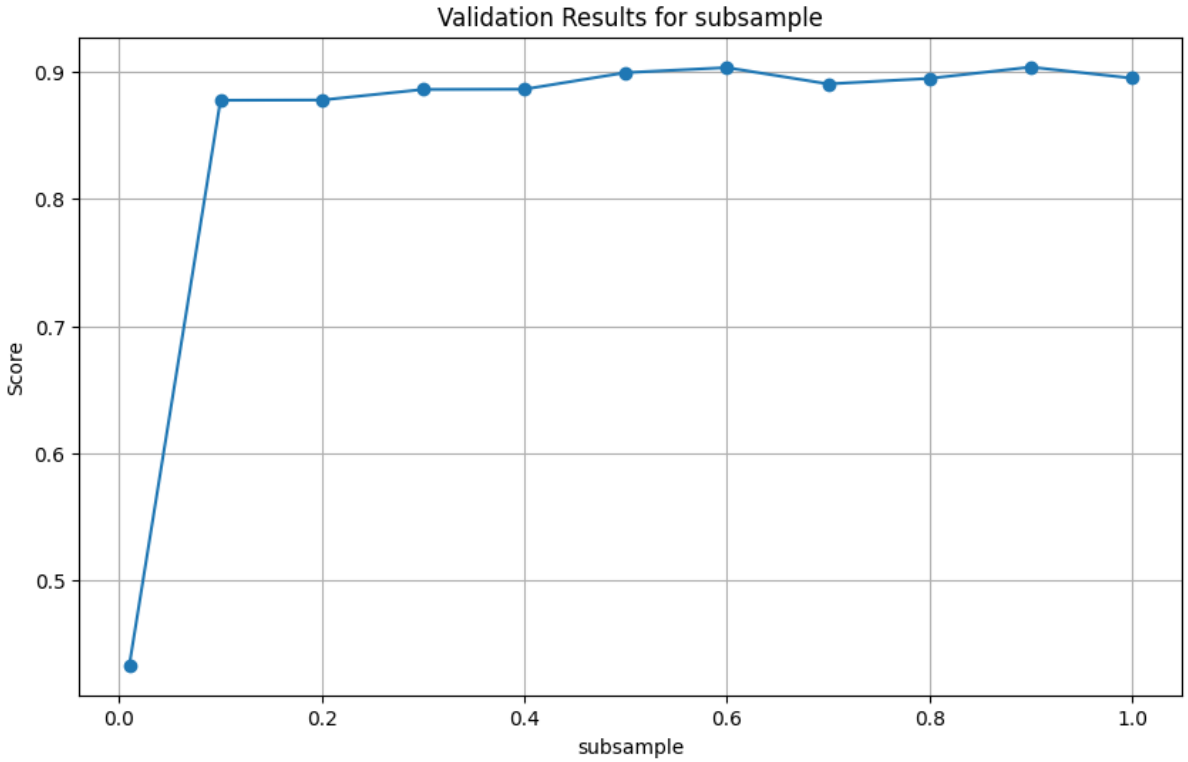


Figure 76. Grid Search XGB - Subsample - Geodesic Distance

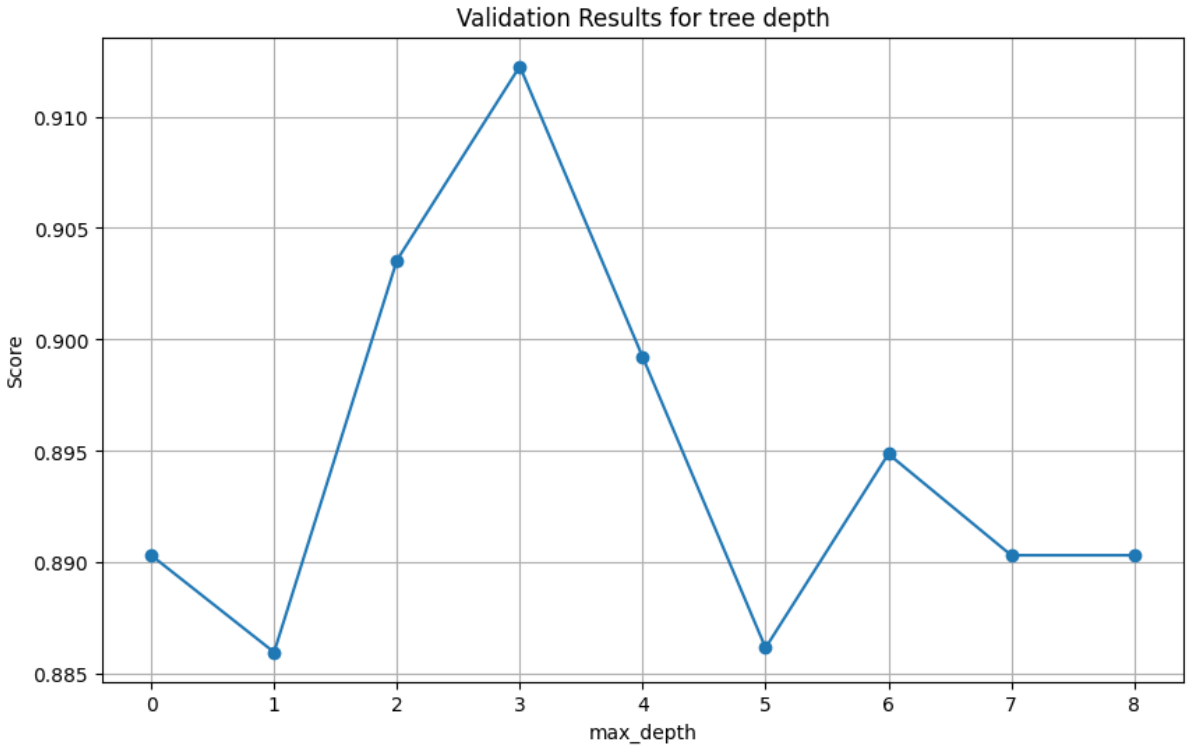


Figure 77. Grid Search XGB - Max depth - Geodesic Distance

After finding optimal areas, a final grid search was run with the following parameters.

```
param_grid = {
    'n_estimators': list(range(125, 150)),
    'subsample': [0.6],
    'colsample_bytree' : [0.4],
    'learning_rate': [0.3],
    'max_depth':[3]}
```

The best parameters were found for **colsample = 0.4, learning_rate=0.3, max_depth = 3, n_estimators=126, subsample = 0.6 and a validation accuracy of 0.8903**. The model was then retrieved and evaluated on the test set. The results are illustrated below.

Table 17. Tuned XGBoost performance - Geodesic Distance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.87	0.98	0.92	0.88
Bus	0.91	0.84	0.87	
Walk	0.85	0.80	0.82	
Macro Average	0.88	0.87	0.87	

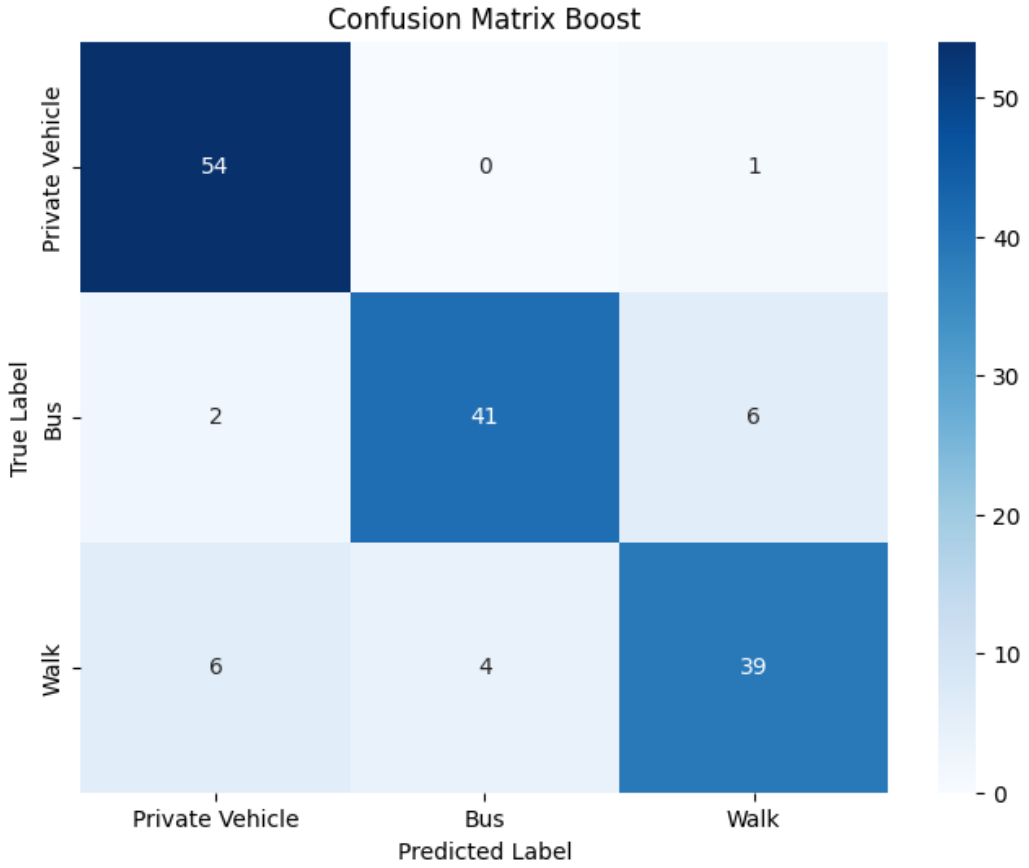


Figure 78. Confusion Matrix Tuned XGBoost - Geodesic Distance

For the tuned model, recall for both bus and private vehicle increased at 0.84 and 0.98 respectively. Precision has increased for both private vehicle and bus from 0.85 and 0.87 up to 0.87 and 0.91 respectively. In contrast, precision for walk decreased from 0.87 to 0.85, while recall remained the same. The model misclassified 6 cases each as walk and private vehicle while their true label was bus and walk respectively. The feature importance was also retrieved for the tuned model. The results are depicted below.

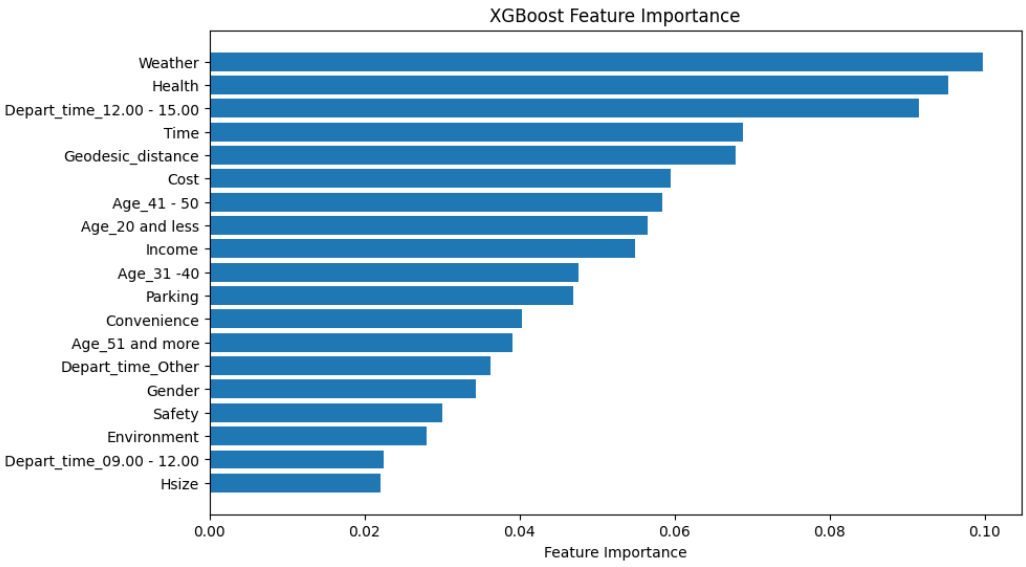


Figure 79. XGBoost feature importance - Geodesic Distance

Surprisingly, the model diverges significantly from the previous models as geodesic distance has dropped at 5th among the most important features. First are now Weather and Health features followed by whether the commute was between 12.00-15.00 or not. The least important features are household size and whether the commute time was between 09.00 – 12.00 or not. While the model seemed to use all the features for the construction of the individual trees, not a concrete decision can be made on the importance below the third place and Depart time (12.00-15.00), since the values for feature importances are relatively close to each other and more data are required. The table below depicts the metrics for both models (Distance vs Geodesic).

Table 18 XGBoost (Distance vs Geodesic Distance)

	XGBoost (Distance)			XGBoost (Geodesic distance)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.95	0.96	0.88	0.87	0.91	0.85
Recall	0.96	0.88	0.94	0.98	0.84	0.80
F1-score	0.95	0.91	0.91	0.92	0.87	0.82
Accuracy	0.93			0.88		
AUC-Macro	0.98			0.96		

Utilizing Distance instead of Geodesic distance, demonstrates a better overall performance for XGBoost in almost every metric for the individual classes. The only exception is to recall for private vehicle class, decreased from 0.98 to 0.96, though precision is significantly higher, 0.95 compared to 0.87. The AUC macro score is also higher for the first model demonstrating a better predictive performance.

3.2.7. Stacked Model

The last model applied was a Stacked Model. Stacking occurs when multiple models are combined to create an enhanced classifier. For the stacked model to be constructed, base classifiers and a “meta” learner need to be defined. The tree-based models (Decision Tree, Random Forest and XGBoost) were used as the base estimators while Logistic Regression (one vs rest approach) was used as the final estimator. The StackingCVClassifier of mlxtend library was used to create the model. As a first experimentation the default values for all estimators were used. The configuration of the model is illustrated below.

```
base_classifier1 = RandomForestClassifier(random_state=1,n_jobs=-1)
base_classifier2 = xgb.XGBClassifier(objective='multi:softmax',
num_class=3, random_state=42)
base_classifier3 = DecisionTreeClassifier(random_state=2)
meta_classifier = LogisticRegression(multi_class='ovr')
stacking_classifier = StackingCVClassifier(
    classifiers=[base_classifier1, base_classifier2, base_classifier3],
    meta_classifier=meta_classifier,
    cv=10,
    stratify=True,
    random_state=10
)
```

The model was initially fitted on the training set and was evaluated on the test set afterwards. The results from the test set are illustrated on the classification report and the confusion matrix below.

Table 19. Stacked model performance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.83	0.96	0.89	0.88
Bus	0.92	0.73	0.82	
Walk	0.90	0.92	0.91	
Macro Average	0.88	0.87	0.87	

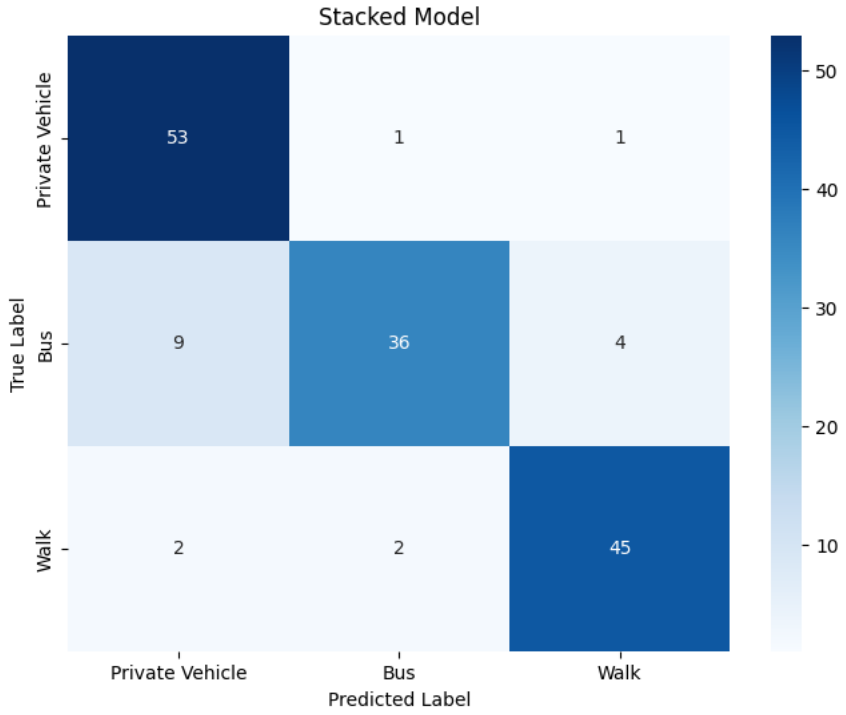


Figure 80. Confusion Matrix - Stacked model

The overall accuracy of the model was 0.88. For "Private Vehicle," the model achieves high recall (0.96) and a precision of 0.83. For bus label, precision is at 0.92, while recall is low at 0.73 retrieving 36 out of 49 true labels. For walk, precision and recall are 0.90 and 0.92 respectively, retrieving 45 out of 49 relevant instances. The highest misclassification occurs between bus and private vehicle, where the model predicted 9 instances as private vehicle while the true label was bus.

The second experimentation was to use the tuned models from the previous chapters. The configuration of the new stacked model is illustrated below.

```
base_classifier1 = RandomForestClassifier(random_state=1,n_jobs=-1, max_depth=7, max_features=10, n_estimators=75)
base_classifier2 = xgb.XGBClassifier(objective='multi:softmax', num_class=3, random_state=1,
    n_estimators=50,
    subsample=0.7,
    colsample_bytree=0.8,
    learning_rate=0.1,
    max_depth=8)
base_classifier3 = DecisionTreeClassifier(random_state=21, max_depth=5, min_samples_leaf = 3, min_samples_split= 2)
meta_classifier = LogisticRegression(multi_class='ovr')
stacking_classifier = StackingCVClassifier(
    classifiers=[base_classifier1, base_classifier2, base_classifier3],
    meta_classifier=meta_classifier,
    cv=10,
    stratify=True,
    random_state=10
)
```

The model was initially fitted on the training set and was evaluated on the test set afterwards. The results from the test set are illustrated on the classification report and the confusion matrix below.

Table 20. Stacked Model performance (optimal estimators)

	Precision	Recall	F1	Accuracy
Private Vehicle	0.96	0.93	0.94	0.90
Bus	0.86	0.86	0.86	
Walk	0.86	0.90	0.88	
Macro Average	0.89	0.89	0.89	

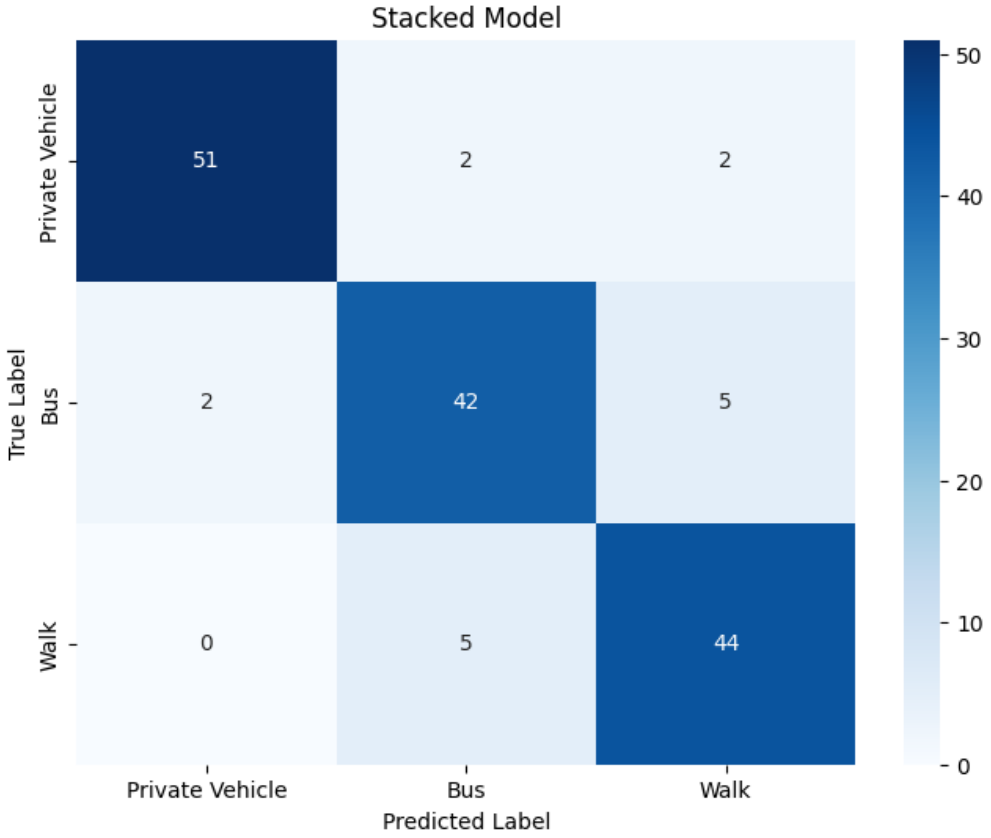


Figure 81. Confusion Matrix Stacked (optimal estimators)

The overall accuracy of the new stacked increased from 0.88 to 0.90. There is a trade off in recalls for the classes. For bus it significantly increased from 0.73 to 0.86, while for private vehicle and walk it slightly dropped at 0.93 and 0.90 from 0.96 and 0.92 respectively. Precision for private vehicle also increased from 0.83 to 0.96, while for bus and walk it dropped at 0.86. The biggest misclassification occurs between walk and bus, where the model predicted 5 instances as walk when the true label was bus and vice versa. The coefficients of the meta estimator (Logistic Regression) were retrieved for the model. The output is illustrated below.

Table 21. Meta estimator coefficients

	Random Forest	XGBoost	Decision Tree
Private vehicle	-2.50	-0.96	-0.85
Bus	0.65	-0.31	-0.35
Walk	1.73	1.72	1.39

Generally, the coefficients represent the change in log odds for predicting that class for a one-unit increase. Though, while in a standard Logistic regression model those coefficients are assigned to the features of the model, in the stacked version those values are assigned to each base estimator. Higher values indicate a higher change in the prediction impact of the class. For instance, an increase in the Random Forest prediction, for private vehicle, has a negative impact in predicting that class and the

magnitude of that impact is 2.5. Additionally, an increase in XGboost prediction for class walk, increase the impact for perdicitng that class by 1.72. The same procedure could be implemented for every coefficient.

As a second experimentation, again geodesic distance was swapped with Distance and a new stacked model was constructed using the optimal parameters of models utilizing geodesic distance. The configuration for the model is illustrated below.

```
base_classifier1 = RandomForestClassifier(random_state=1,n_jobs=-1, max_depth=5, max_features=14, n_estimators=141)
base_classifier2 = xgb.XGBClassifier(objective='multi:softmax', num_class=3, random_state=42,
                                     colsample_bytree= 0.4, learning_rate= 0.3, max_depth= 3, n_estimators= 126, subsample= 0.6)
base_classifier3 = DecisionTreeClassifier(random_state=21, max_depth=6, min_samples_leaf = 2, min_samples_split= 2)
meta_classifier = LogisticRegression(multi_class='ovr')
stacking_classifier = StackingCVClassifier(
    classifiers=[base_classifier1, base_classifier2, base_classifier3],
    meta_classifier=meta_classifier,
    cv=10,
    stratify=True,
    random_state=1
)
```

Table 22. Stacked - Geodesic Distance

	Precision	Recall	F1	Accuracy
Private Vehicle	0.87	0.96	0.91	0.86
Bus	0.87	0.80	0.83	
Walk	0.85	0.82	0.83	
Macro Average	0.86	0.86	0.86	

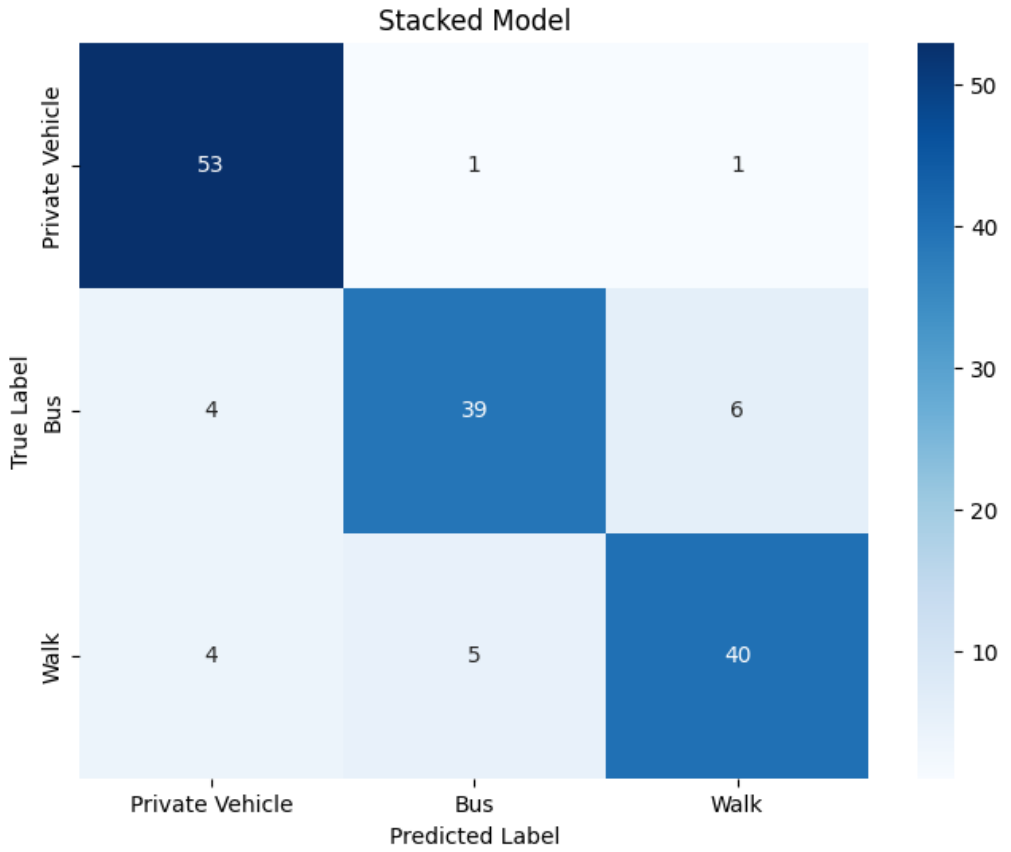


Figure 82. Confusion Matrix Stacked - Geodesic Distance

The overall accuracy of the model was 0.86. For "Private Vehicle," the model achieves high recall (0.96) and a precision of 0.87. For "Bus" category, precision is at 0.87, while recall is lower at 0.80. For walk, precision and recall are also lower at 0.85 and 0.82 respectively. The highest misclassification occurs between walk and bus classes as the model predicted 6 cases as walk while the true label was bus. The overall performance of the model dropped by swapping distances.

The table below depicted the metrics of the models utilizing different distance metrics.

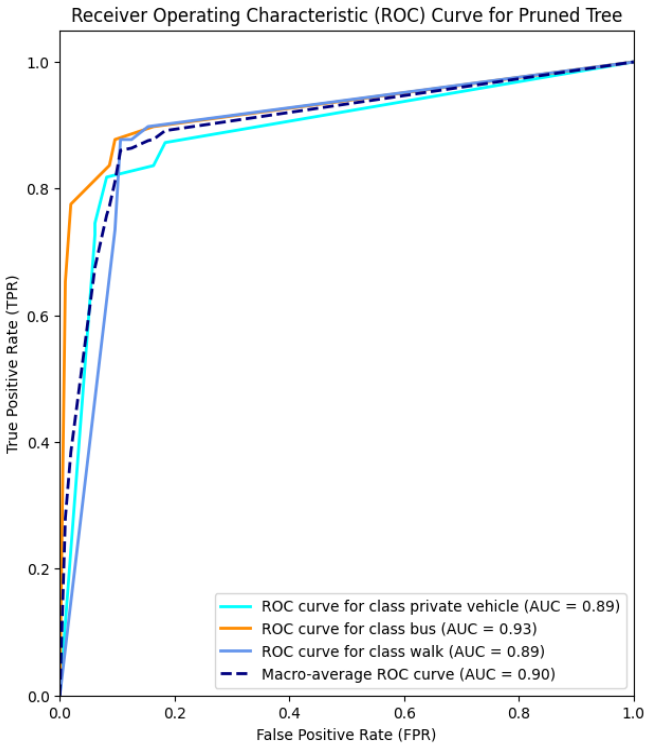
Table 23. Stacked Model (Distance vs Geodesic Distance)

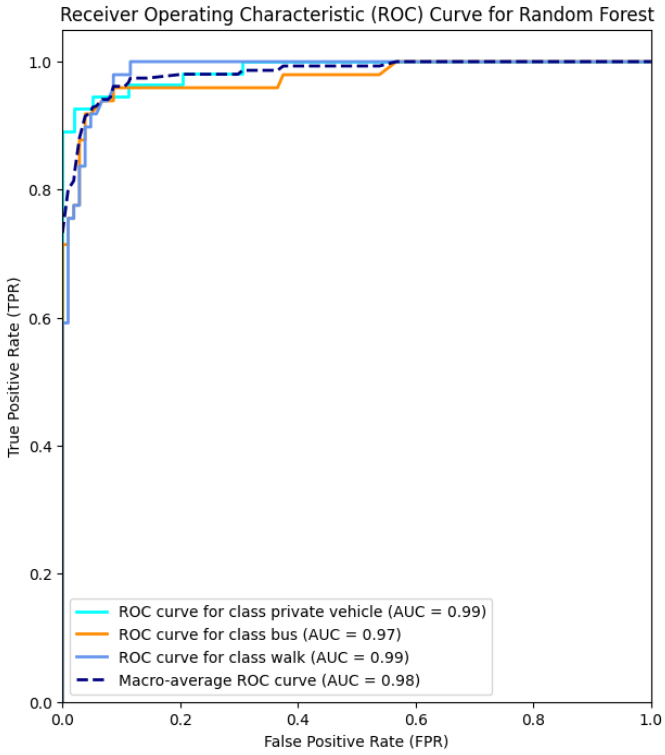
	Stacked (Distance)			Stacked (Geodesic distance)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.96	0.86	0.86	0.87	0.87	0.85
Recall	0.93	0.86	0.90	0.96	0.80	0.82
F1-score	0.94	0.86	0.88	0.91	0.83	0.83
Accuracy	0.90			0.86		
AUC-Macro	0.9304			0.9265		

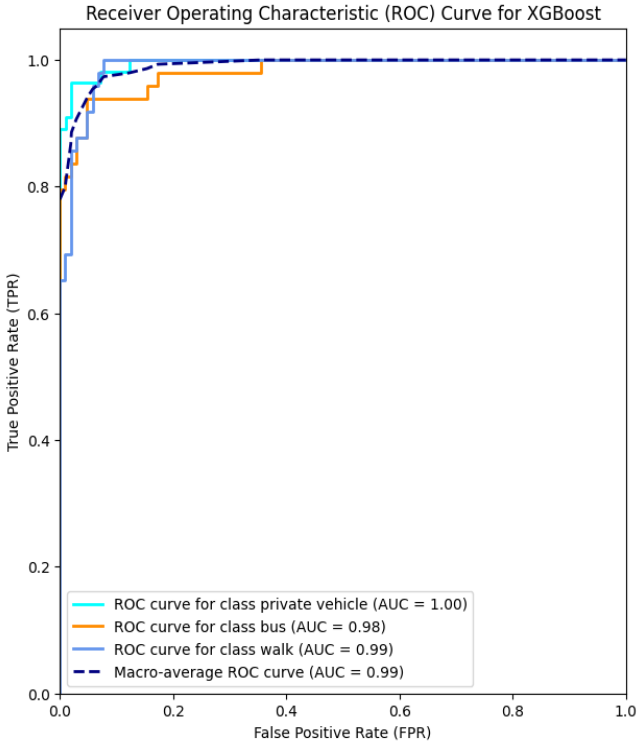
The first model demonstrates a more balanced performance across the classes, while the auc macro score is slightly higher that the model with Geodesic distance, indicating a better predictive capability.

3.2.8. Model Comparison

Before assessing the model performance, the ROC curves for the best models were constructed. Although these curves are typically created for binary classification scenarios, in a multi-classification task, curves are generated for each individual class using a "One vs Rest" approach. For example, in the case of the "private vehicle" class, the true positive rate signifies the ratio of correctly predicted observations belonging to that class. Conversely, the false positive rate for "private vehicle" indicates the ratio of observations falsely predicted as belonging to that class when they actually belong to any of the other classes (bus or walk). The metric representing the model's performance is the area under the curve (AUC), calculated for each class individually in a multiclass task. The AUC macro score is then derived by averaging the AUC scores for each class. This score ranges from 0 to 1, providing an overall assessment of the model's performance. The ROC curves of the models utilizing Distance feature are presented below.







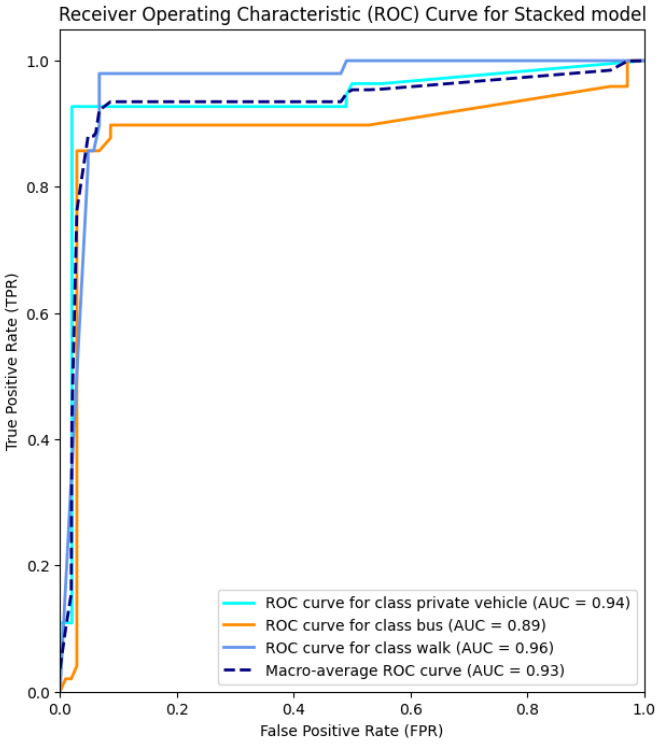


Figure 83. ROC Curves - Distance

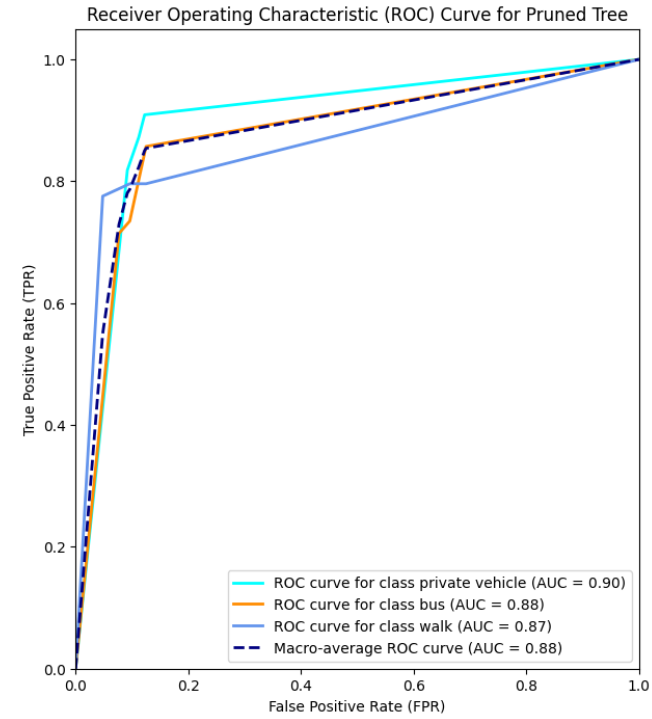
After applying the 4 models to the training set and evaluating their predicting performance on the test set, while also constructing their ROC curves, a concrete decision must be made on which model performs the best and can be trusted for prediction on new unseen data. For that purpose, a comparative table was constructed highlighting the averages of Precision, Recall, F-score, AUC between classes and the overall accuracy of the best models that were constructed. The models below all use the Distance feature.

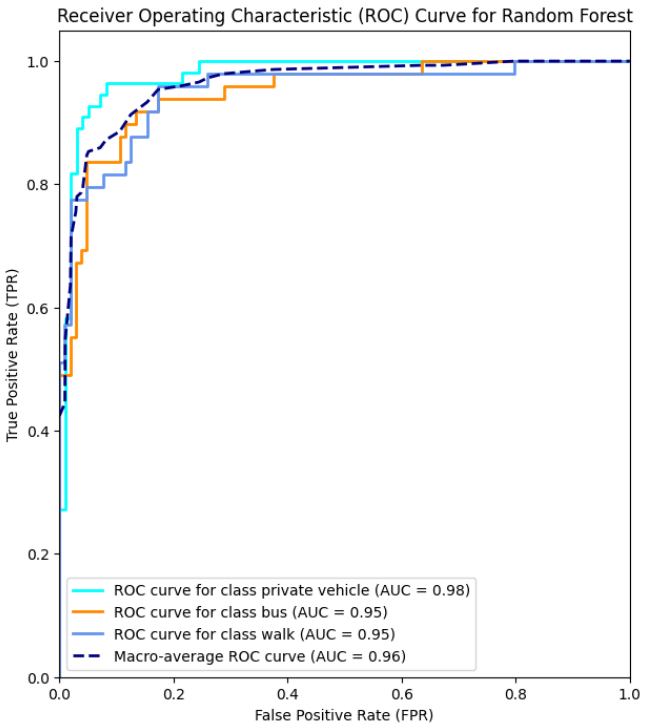
Table 24. Model Comparison - Distance

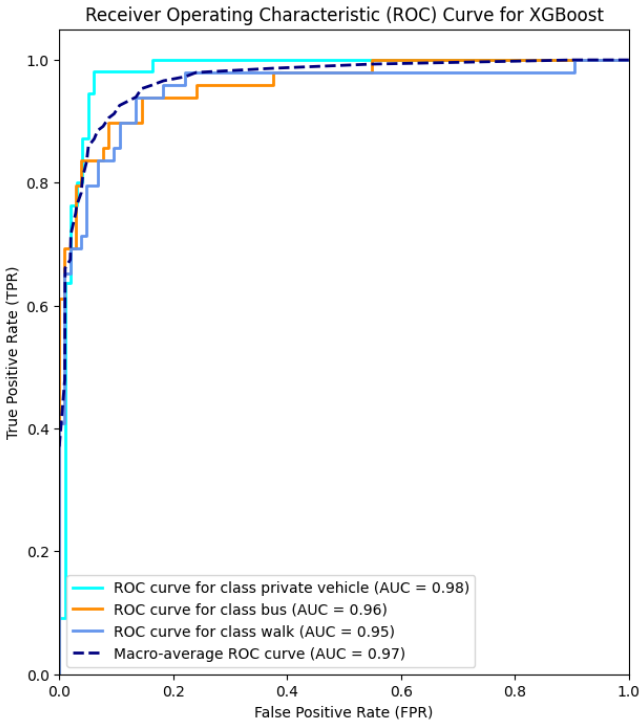
Models with Distance	**Decision Tree**	**Random Forest**	**XGBoost**	**Stacked Model**
Precision (Average)	0.83	0.92	**0.93**	0.89
Recall (Average)	0.83	0.91	**0.93**	0.89
F-Score (Average)	0.83	0.91	**0.93**	0.89
AUC (Average)	0.90	0.98	**0.985**	0.93
Accuracy	0.83	0.92	**0.93**	0.90

From the table above it is evident that XGBoost and Random Forest demonstrate the best overall performance for Precision, Recall and F scores. They also demonstrate the highest macro AUC above 0.98. XGBoost demonstrates a slightly better performance as AUC for XGBoost is 0.986 compared to 0.980 for Random Forest. Hence, XGBoost is the model to be used for new unseen data using the Distance feature.

The roc curves were also generated for the models that utilize Geodesic Distance. The plots are illustrated below along with the model comparative table.







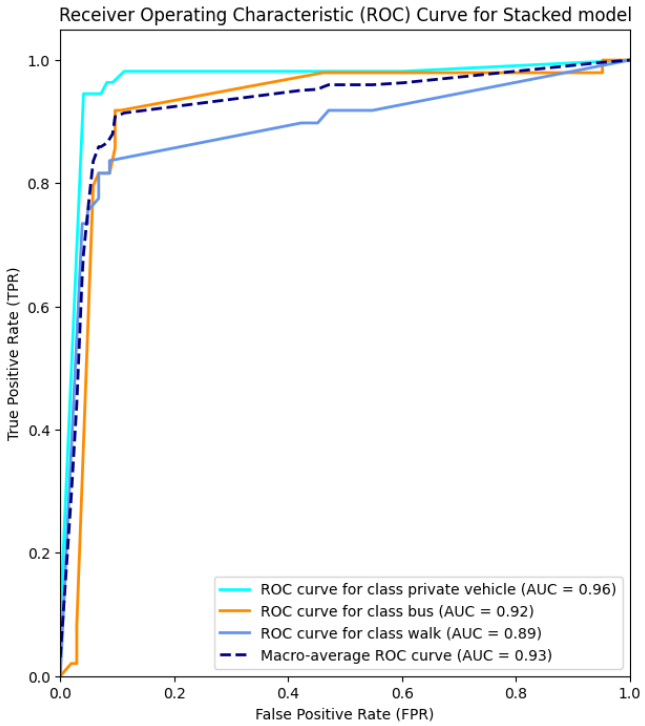


Figure 84. ROC Curves - Geodesic Distance

Table 25. Model Comparison - Geodesic Distance

Models with Geodesic Distance	**Decision Tree**	**Random Forest**	**XGBoost**	**Stacked Model**
Precision (Average)	0.83	0.86	**0.88**	0.86
Recall (Average)	0.82	0.85	**0.87**	0.86
F-Score (Average)	0.82	0.85	**0.87**	0.86
AUC (Average)	0.88	0.96	**0.96**	0.926
Accuracy	0.82	0.86	**0.88**	0.86

From the table above it is evident that again XGBoost demonstrates the best overall performance for Precision, Recall and F scores. It also demonstrates the highest macro-AUC at 0.96. While all the metrics dropped compared to the model utilizing the Distance feature, XGBoost is the model to implement for predictions on new unseen data regardless of distance metric used.

3.2.9. Feature reduction and re-evaluation

In previous chapters, the models were trained and evaluated, while 19 features were used in total, including those that were one hot encoded. The models were re-evaluated after dropping most of the features. By observing the plots of the model importance, the 7 most influential features for tree construction were selected. Those features are **Income, Time, Convenience, Cost, Health, Weather and Distance**. The new models were initially trained on the training set. Afterwards, a grid search was executed to identify the new optimal parameter values and the models were evaluated on the test set afterwards.

A) Decision tree

For decision tree, a grid was run with the following parameters:

```
param_grid = {
   'max_depth': [2, 3, 4, 5, 6],
```

'min_samples_split': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],

'min_samples_leaf': [1, 2, 3, 4, 5, 6, 7]

The best results were for **max_depth = 6, min_samples_leaf=1 and min_samples_split=3 with a validation accuracy of 0.868**. The table below illustrates the performance of the new tree on the test set compared to the one using all the 19 features.

Table 26. Pruned Tree (7 vs 19 features)

	Tree (7 features)			Tree (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.81	0.91	0.78	0.82	0.89	0.80
Recall	0.85	0.82	0.82	0.82	0.80	0.88
F1-score	0.83	0.86	0.80	0.82	0.84	0.83
Accuracy	0.83			0.83		
AUC-Macro	0.88			0.90		

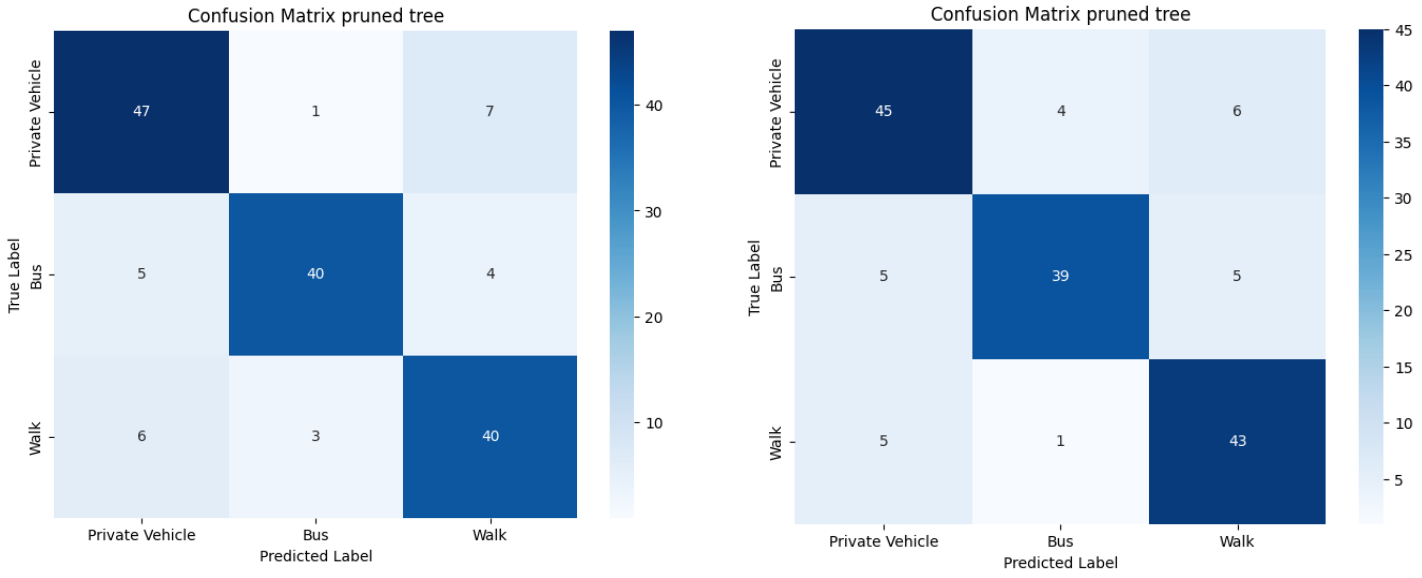


Figure 85. Confusion Matrix Tree (7 vs 19 features)

It is observed that feature reduction had minor changes on the metrics for the decision tree. Recall for bus increased from 0.80 to 0.82, while for private vehicle also increased from 0.82 to 0.85. In contrast, recall for walk dropped from 0.88 to 0.82. Precision only increased for bus label, from 0.89 to 0.91, while it slightly dropped for private vehicle and walk. AUC macro score is slightly higher for the tree utilizing all of 19 features, demonstrating slightly better performance.

The same procedure was executed by swapping Distance with geodesic distance. A new grid was run to find the optimal values. The results were best for **max_depth = 5, min_sampels_leaf=1, min_samples_split=2 and a validation accuracy of 0.845.** The table below shows the results of the new tree compared to the tree utilizing all of 19 features.

Table 27. Pruned Tree - Geodesic Distance (7 vs 19 features)

	Geodesic Tree (7 features)			Geodesic Tree (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.79	0.87	0.81	0.81	0.78	0.88
Recall	0.96	0.69	0.78	0.87	0.82	0.78
F1-score	0.87	0.77	0.79	0.84	0.80	0.83
Accuracy	0.82			0.82		
AUC-Macro	0.89			0.88		

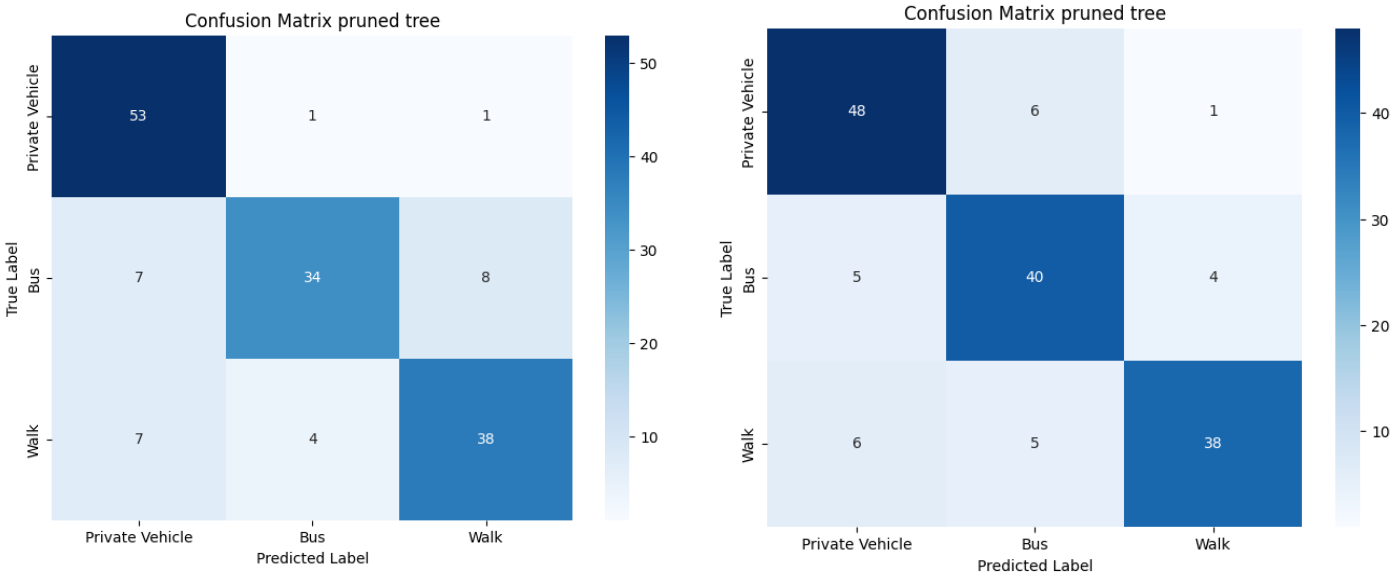
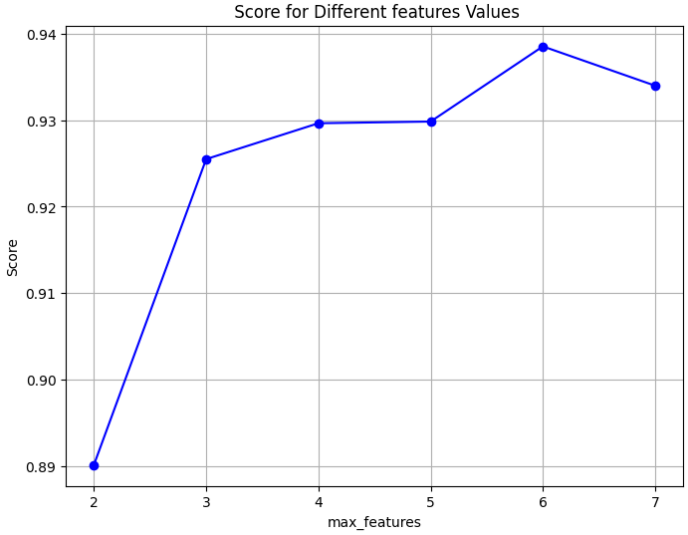


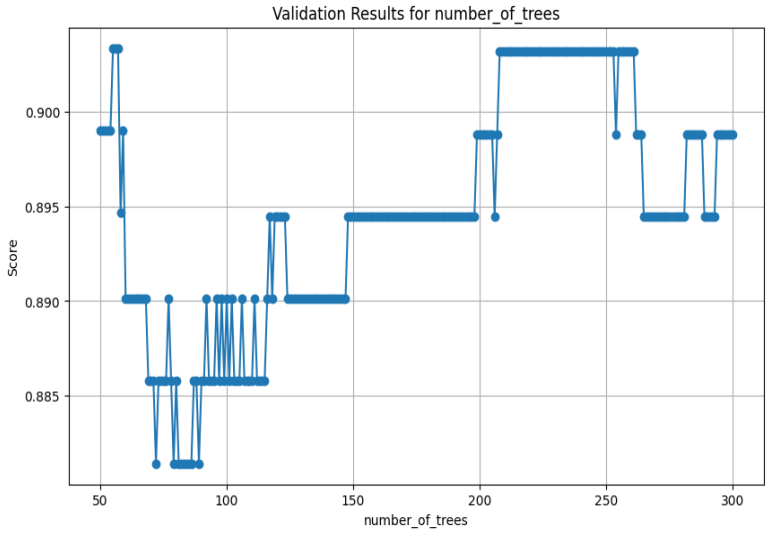
Figure 86. Confusion Matrix Tree - Geodesic Distance (7 vs 19 features)

For private vehicle class, recall increased from 0.87 to 0.96. In contrast, recall for bus dropped from 0.82 to 0.69. Recall for walk remained the same at 0.78. Precision increased for bus, from 0.78 to 0.87 while it dropped for walk and private vehicle at 0.81 and 0.79 respectively. The overall accuracy remained the same at 0.82. While the auc macro is slightly higher at 0.89. Overall, feature reduction did not improve the performance of the trees regardless of distance metric. In contrast while for private vehicle more relevant instances were retrieved, for bus recall drops significantly using geodesic distance, as also evident from the confusion matrices.

B) ***Random Forest***

For Random Forest the same parameters were examined as before: the number of trees, max features, and max tree depth. A grid search was run for each parameter individually. The results are illustrated





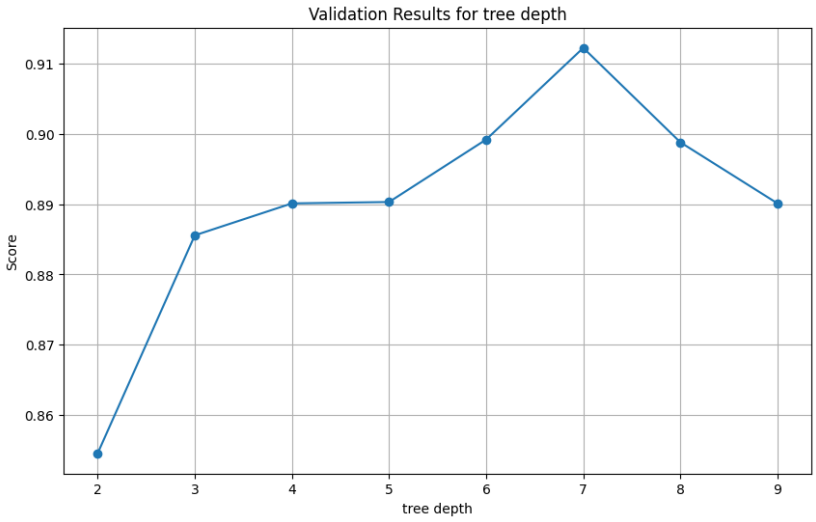


Figure 87. Grid search RF - 7 features

After finding the optimal area, a final grid was run using the following parameter values.

```
param_grid = {
    'n_estimators': list(range(50, 61)),
    'max_features': [6],
    'max_depth' : [7]}
```

The best results are for **max depth =7, max features =6, number of trees = 55 and a validation accuracy of 0.934**. The model was then evaluated on the test set. The table below depicts the differences of the two Random Forest models (7 vs 19 features).

Table 28. Random Forest (7 vs 19 features)

	Random Forest (7 features)			Random Forest (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.96	0.98	0.87	0.96	0.95	0.84
Recall	0.93	0.92	0.96	0.93	0.86	0.96
F1-score	0.94	0.95	0.91	0.94	0.90	0.90
Accuracy	0.93			0.92		
AUC-Macro	0.985			0.983		

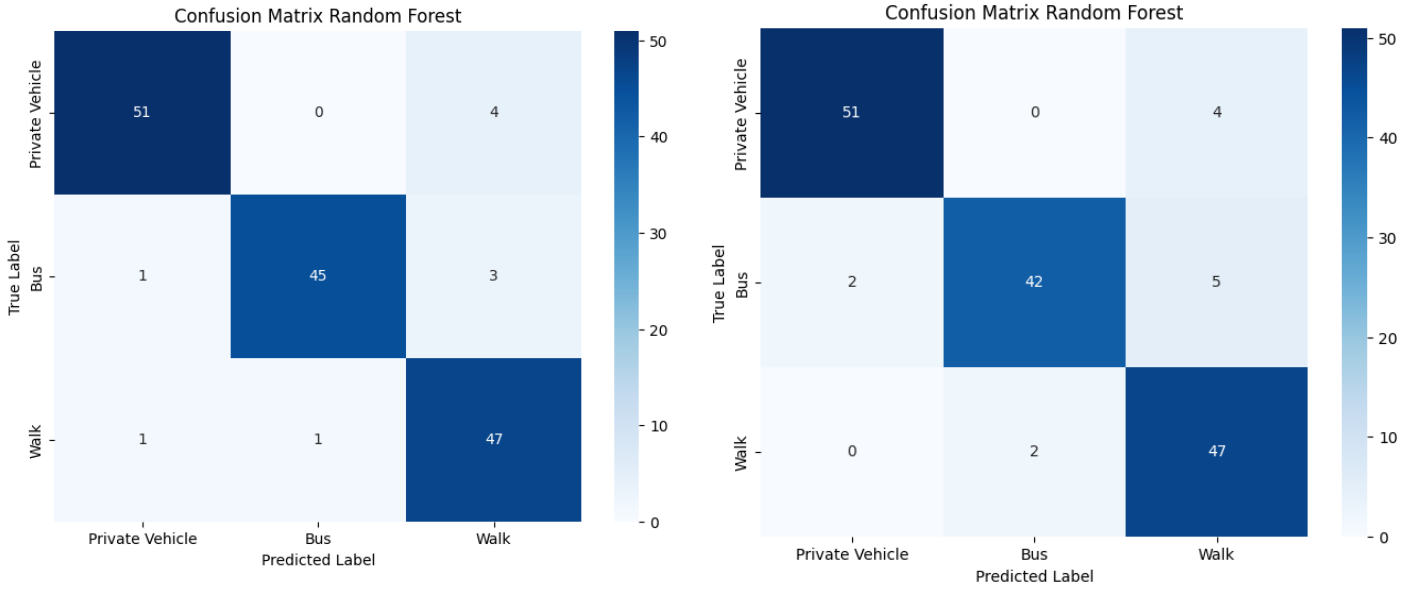


Figure 88. Confusion Matrix RF (7 vs 19 features)

For the new model, all metrics for private vehicle remained the same. Both recall and precision increased at 0.92 and 0.98 respectively. Precision also increased for walk, from 0.84 to 0.87 while recall remained the same at 0.96. The overall accuracy increased from 0.92 to 0.93 while auc macro slightly improved from 0.983 to 0.985. Overall, by reducing the features of the model, increased the performance of Random Forest.

Distance was again swapped with Geodesic distance. The new optimal parameters are illustrated on the plots below.

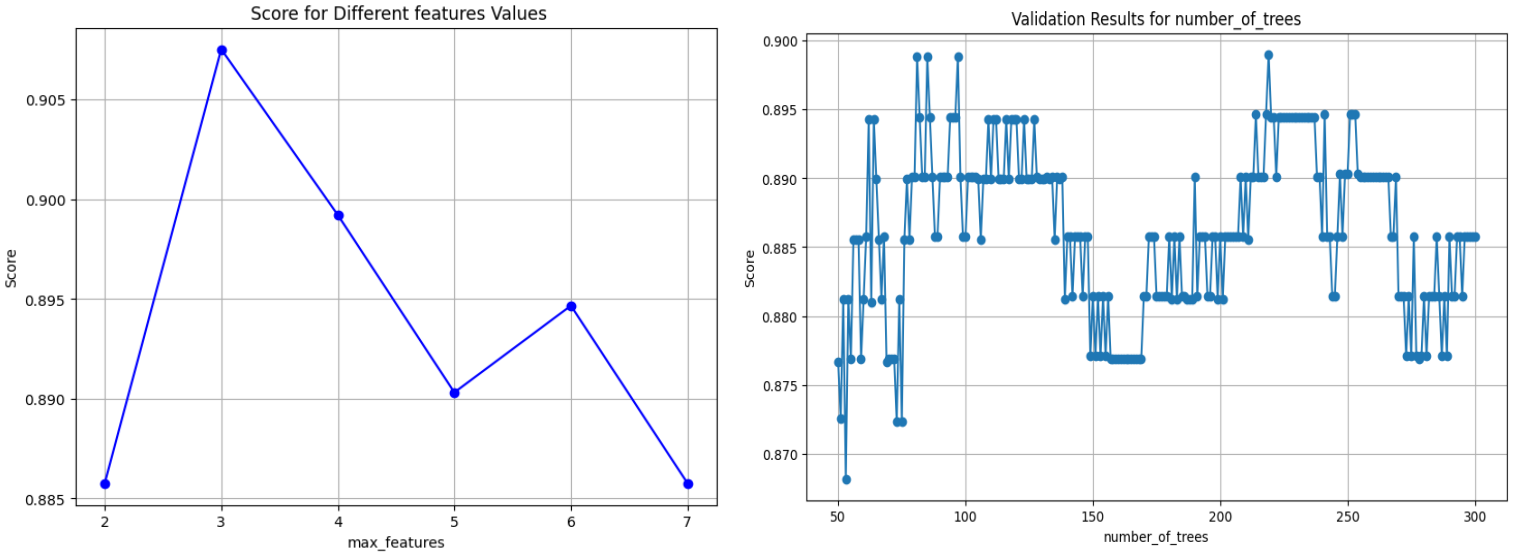


Figure 89. Grid Search - Geodesic Distance - 7 features

After finding the optimal area, a new grid search was run with the following parameter values.

param_grid = {

'n_estimators': list(range(60, 101)),

'max_features': [3]}

The best results are for max features = 3, number of trees = 68 and a validation accuracy of 0.91. The model was then evaluated on the test set. The results are illustrated below compared to the model using all 19 features.

Table 29. Random Forest – Geodesic Distance (7 vs 19 features)

	Geodesic Random Forest (7 features)			Geodesic Random Forest (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.90	0.89	0.84	0.88	0.95	0.91
Recall	0.95	0.84	0.84	0.88	0.78	0.83
F1-score	0.92	0.86	0.84	0.80	0.84	0.82
Accuracy	0.88			0.86		
AUC-Macro	0.96			0.96		

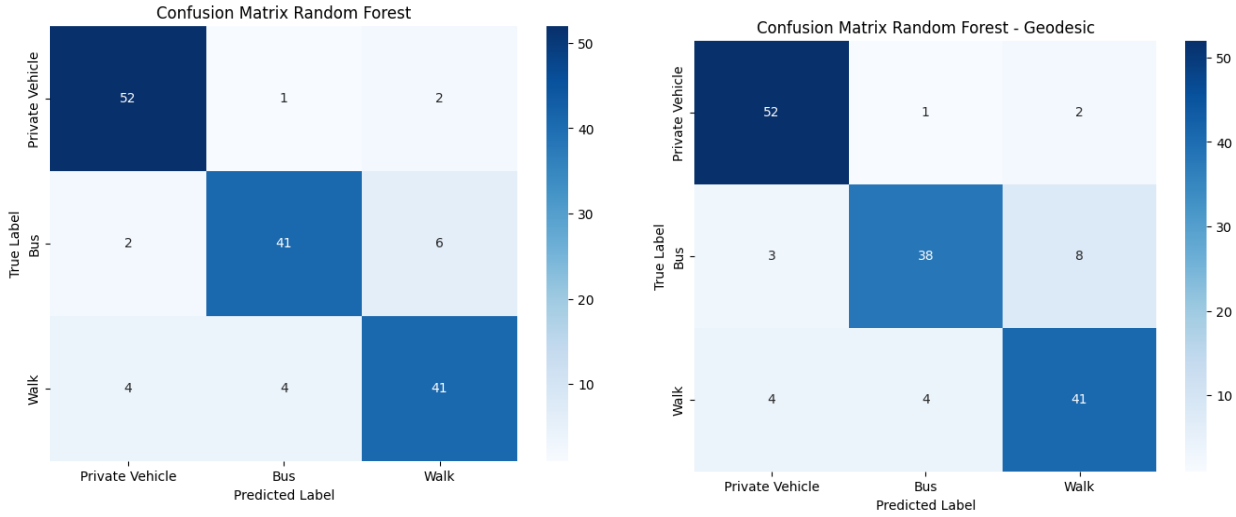


Figure 90. Confusion Matrix - Geodesic Distance (7 vs 19 features)

For private vehicle, recall increased from 0.88 to 0.95. For walk it also increased from 0.83 to 0.84, while for bus it also increased from 0.78 to 0.84. Precision increased for private vehicle from 0.88 to 0.90 while it dropped for the other two classes. The overall accuracy of the model increased from 0.86 to 0.88 while the auc macro score remained approximately the same. By reducing the features, the Random Forest model (using geodesic distance) demonstrates a more balanced performance, also retrieving more relevant instances for each class.

C) *XGBoost*

For XGBoost, the same parameters were tuned as before. The values that were configured in grid search are depicted below

'n_estimators': list(range(100, 151)),

'subsample': [0.3],

'colsample_bytree' : [0.9],

'learning_rate': [0.2],

'max_depth':[2]}

The best result was for **number of trees =100, max depth=2, learning rate=0.2, subsample=0.3, colsample =0.9 with a validation accuracy of 0.947**. The comparisons of the two models (7 vs 19 features) on the test set are illustrated below.

Table 30. XGBoost (7 vs 19 features)

	XGBoost (7 features)			XGBoost (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.93	0.96	0.94	0.95	0.96	0.88
Recall	0.98	0.92	0.92	0.96	0.88	0.94
F1-score	0.96	0.94	0.93	0.95	0.91	0.91
Accuracy	0.94			0.93		
AUC-Macro	0.99			0.99		

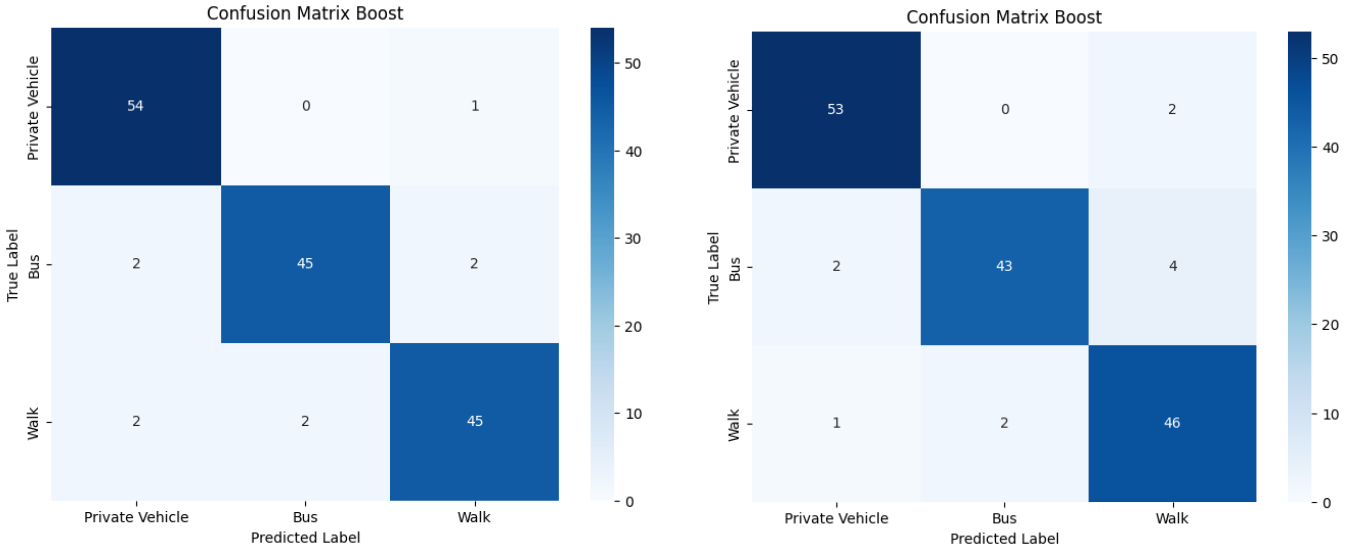


Figure 91. Confusion Matrix XGB (7 vs 19 features)

Recall for private vehicle increased from 0.96 to 0.98 with a trade-off in precision as it dropped at 0.93. Recall also increased for bus from 0.88 to 0.92, while precision remained the same at 0.96. Recall for walk dropped from 0.94 to 0.92 while precision increased from 0.88 to 0.94. The overall accuracy of the model increased from 0.93 to 0.94 while the auc macro remained approximately the same at 0.99. By reducing the features, the model demonstrates a more balanced performance across the classes retrieving also more relevant instances.

Again, the model was evaluated swapping distance with geodesic distance. The new parameter values that were tuned through grid search are depicted below.

```
param_grid = {
    'n_estimators': list(range(50, 61)),
    'subsample': [0.3],
    'colsample_bytree' : [0.8],
    'learning_rate': [0.3],
    'max_depth':[2]}
```

The best parameters were found for **max depth =2, learning rate =0.3, number of trees =50, subsample=0.3, colsample=0.8 and a validation accuracy of 0.89**.

The table below illustrates the differences in the two models (7 features vs 19 features).

Table 31. XGBoost - Geodesic Distance (7 vs 19 features)

	Geodesic XGBoost (7 features)			Geodesic XGBoost (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.88	0.88	0.85	0.87	0.91	0.85
Recall	0.91	0.86	0.84	0.98	0.84	0.80
F1-score	0.89	0.84	0.85	0.92	0.87	0.82
Accuracy	0.87			0.88		
AUC-Macro	0.969			0.967		

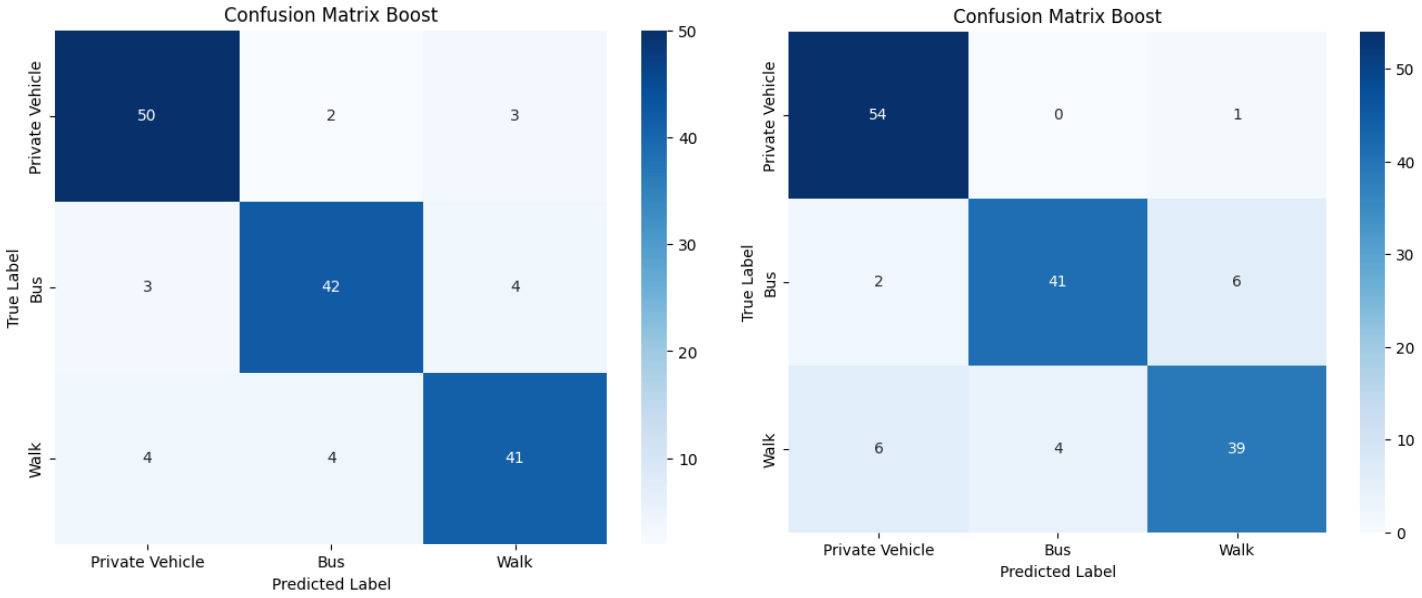


Figure 92. Confusion Matrix XGB - Geodesic Distance (7 vs 19 features)

Recall for private vehicle decreased from 0.98 to 0.91 with a slight increase in precision, from 0.87 to 0.88. Recall for bus increased from 0.94 to 0.96. Recall for walk also increased from 0.80 to 0.84. The overall accuracy slightly drooped from 0.88 to 0.87, while auc macro slightly increased from 0.967 to 0.969. Overall, there is not a significant difference in the model by reducing the number of features.

D) ***Stacked Model***

Finally, new stacked models were constructd using only the 7 available features. The tuned models were used as base estimators (Decision Tree, Random Forest, XGBoost), while Logistic Regression (one vs rest) was used as the meta learner. The table below illustrates the differences in the models utilizing Distance as a feature.

Table 32. Stacked Model (7 vs 19 features)

	Stacked (7 features)			Stacked (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.94	0.86	0.94	0.96	0.86	0.86
Recall	0.91	0.90	0.94	0.93	0.86	0.90
F1-score	0.93	0.88	0.94	0.94	0.86	0.88
Accuracy	0.92			0.90		
AUC-Macro	0.958			0.93		

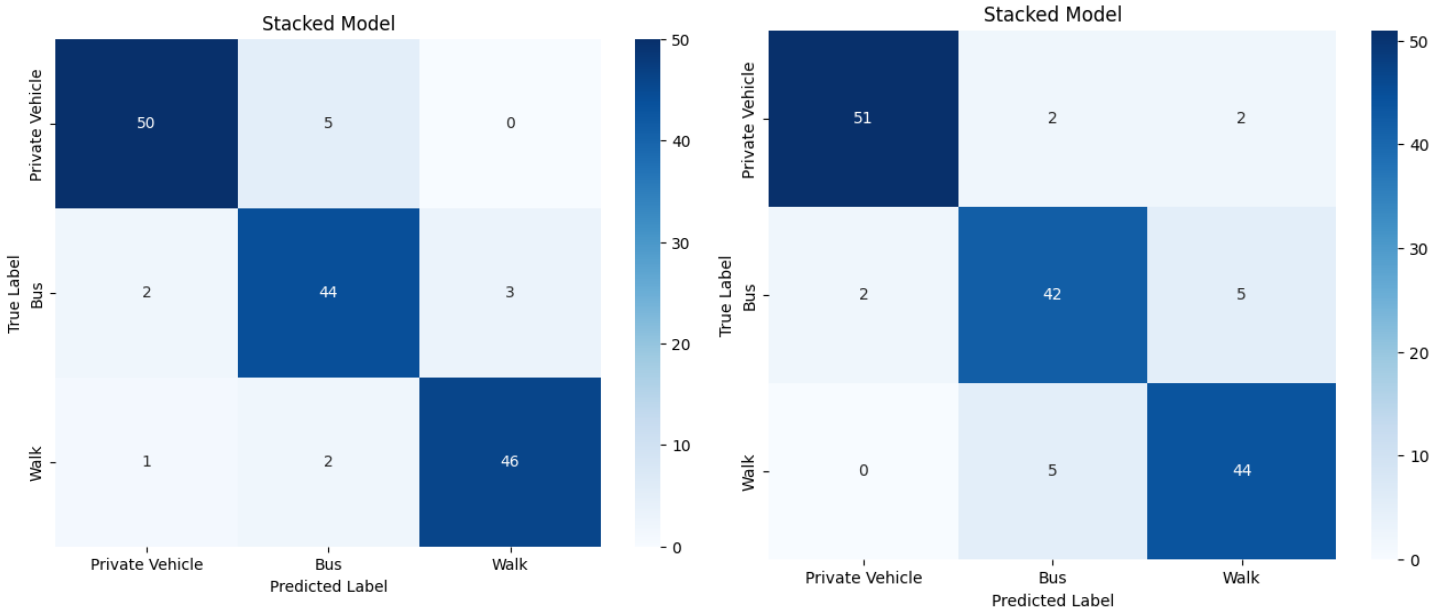


Figure 93. Confusion Matrix - Stacked (7 vs 19 features)

Recall increased for bus, from 0.86 to 0.90 while precision remained at 0.86. Recall for walk also increased from 0.90 to 0.94, while precision increased from 0.86 to 0.94. In contrast, recall for private vehicle decreased from 0.93 to 0.91. The overall accuracy of the model increased from 0.90 to 0.92, while auc macro also increased from 0.93 to 0.958. Overall, by reducing the features the model demonstrates a more balanced performance for the individual classes.

Distance metrics were again swapped and evaluate the new stacked model with reduced number of features. The tuned tree (Decision Tree, Random Forest and XGBoost) using geodesic distance, were used as the base estimators, while Logistic Regression was used as the meta learner. The comparison of the models is illustrated below (Geodesic distance – 7 features vs 19 features).

Table 33. Stacked Model - Geodesic Distance (7 vs 19 features)

	Geodesic Stacked (7 features)			Geodesic Stacked (19 features)		
	Private vehicle	Bus	Walk	Private vehicle	Bus	Walk
Precision	0.86	0.86	0.79	0.87	0.87	0.85
Recall	0.93	0.73	0.84	0.96	0.80	0.82
F1-score	0.89	0.79	0.81	0.91	0.83	0.83
Accuracy	0.84			0.86		
AUC-Macro	0.95			0.926		

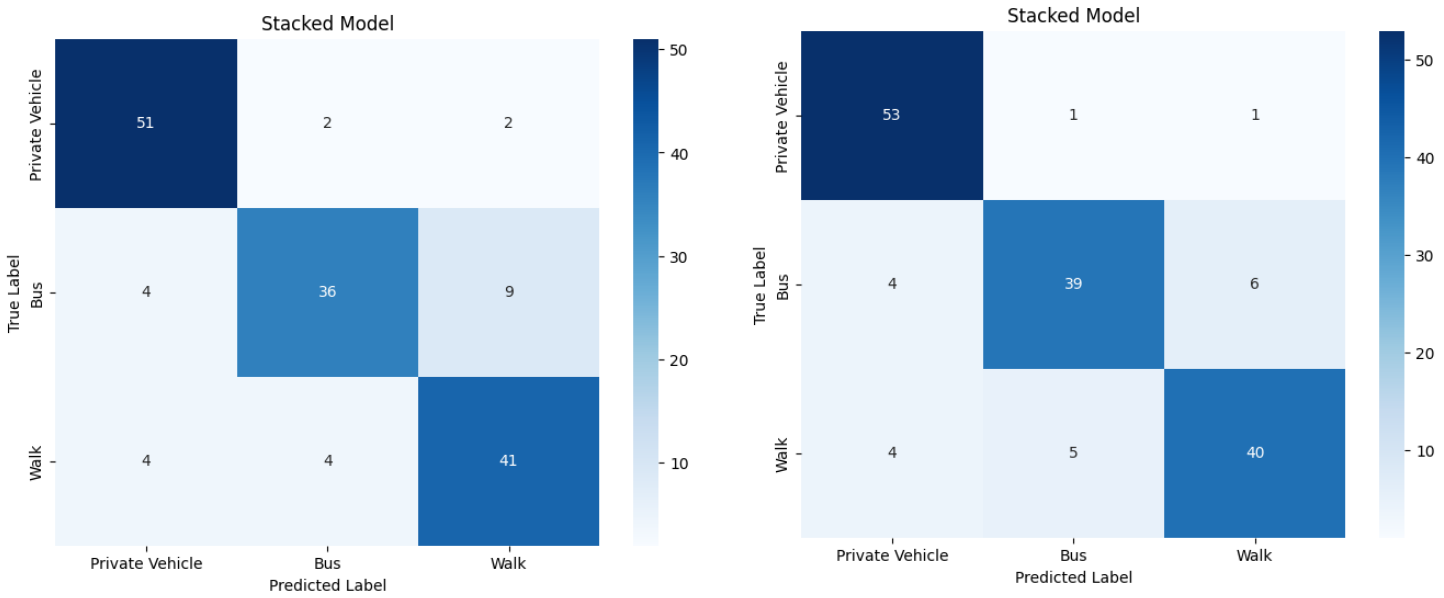
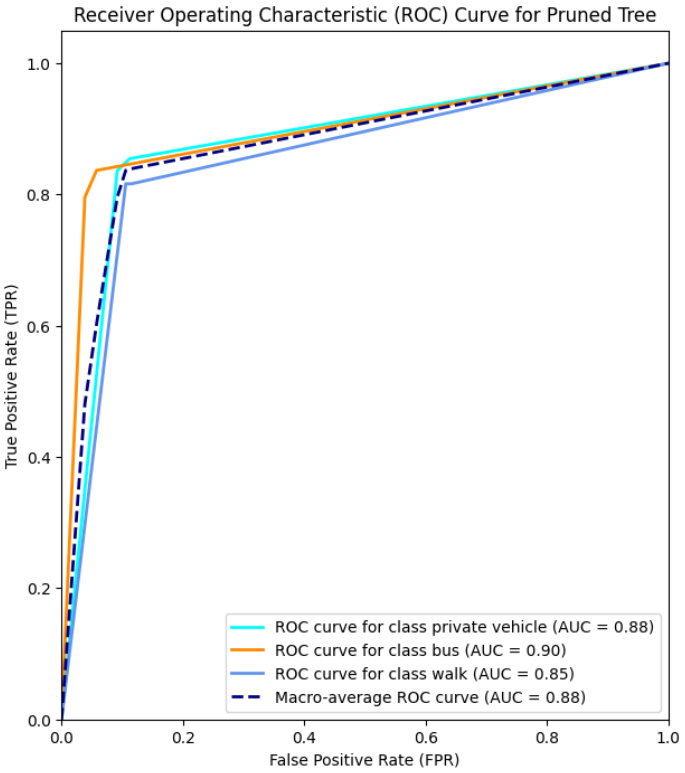
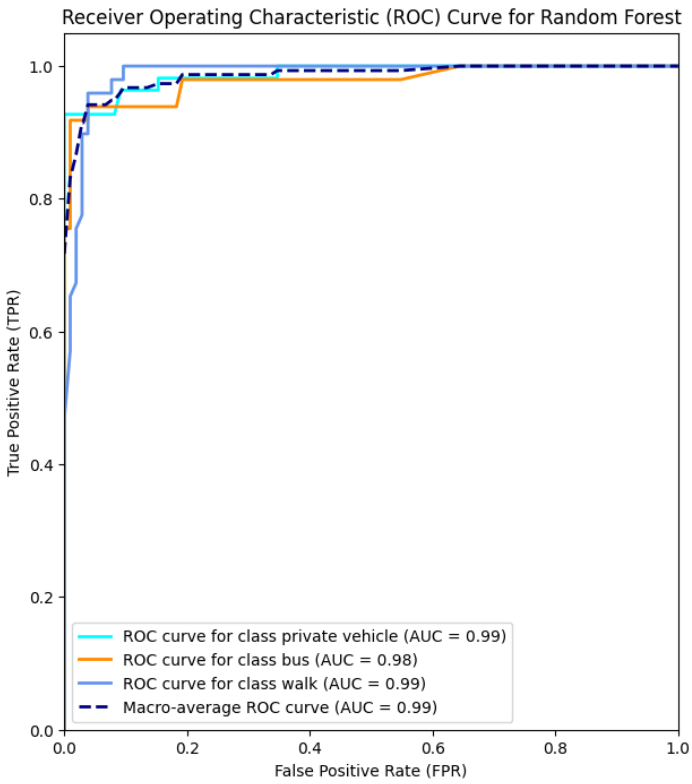


Figure 94. Confusion Matrix Stacked - Geodesic Distance (7 vs 19 features)

Recall for private vehicle reduced from 0.96 to 0.93. Recall for bus also dropped from 0.80 to 0.73. In contrast, recall for walk increased from 0.82 to 0.84. Precision for all individual classes also dropped. The overall accuracy of the model dropped from 0.86 to 0.84. By reducing the features and using geodesic distance, stacked model seems to perform worse compared to the original, since it retrieves lower number of relevant instances, while precision is lower for all classes.

The roc curves were constructed for the new models (using 7 features). The results, along with the model comparison table are illustrated below, initially for the models utilizing Distance metric.





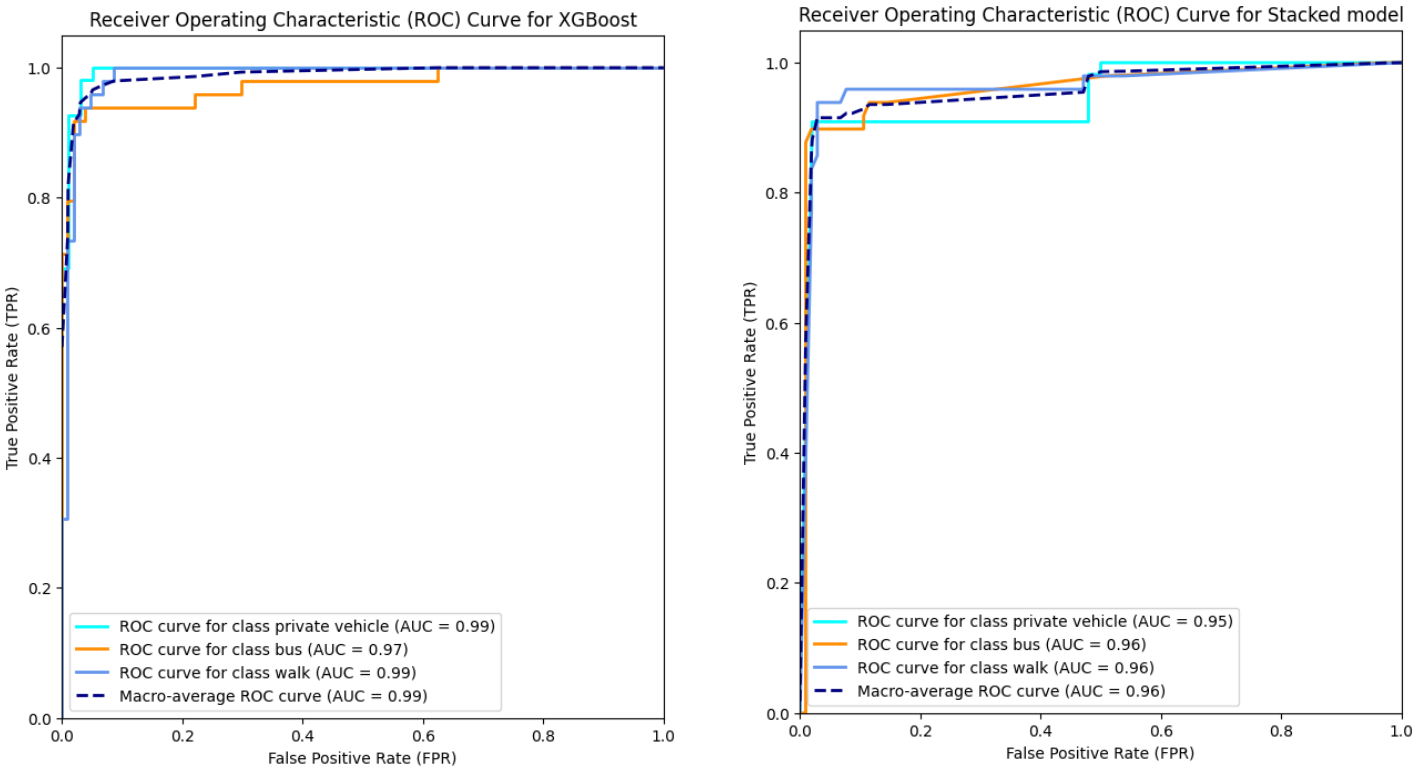


Figure 95. ROC Curves - 7 features

Table 34. Model comparison - 7 features

Models with Distance and Reduced Features	Decision Tree	Random Forest	XGBoost	Stacked Model
Precision (Average)	0.83	0.94	**0.94**	0.91
Recall (Average)	0.83	0.93	**0.94**	0.92
F-Score (Average)	0.83	0.93	**0.94**	0.91
AUC (Average)	0.876	0.985	**0.988**	0.958
Accuracy	0.83	0.93	**0.94**	0.92

The XGBoost model stands out as the top-performing model across precision, recall, and F-score, demonstrating consistently high values. Random Forest closely follows, exhibiting high precision and recall scores, and a high AUC value of 0.985. While Stacked Model also displays commendable performance, XGBoost and Random Forest emerge as particularly better choices for predicting transportation mode utilizing Distance and a reduced set of features. Overall, these models showcase strong predictive capabilities, with XGBoost and Random Forest leading the way.

The same procedure was followed for the models utilizing Geodesic distance. The results are illustrated below.

Table 35. Model comparison - 7 features – Geodesic distance

Models with Geodesic Distance and Reduced Features	Decision Tree	Random Forest	XGBoost	Stacked Model
Precision (Average)	0.82	**0.87**	**0.87**	0.84
Recall (Average)	0.81	**0.87**	**0.87**	0.83
F-Score (Average)	0.81	**0.87**	**0.87**	0.83
AUC (Average)	0.89	**0.96**	**0.96**	0.93
Accuracy	0.82	**0.88**	**0.87**	0.84

Random Forest continues to showcase increased performance, though lower when compared to the previous models. XGBoost closely follows, maintaining a balanced performance across various metrics. Decision Tree and the Stacked Model demonstrate mediocre results, albeit slightly lower than those achieved with distance and reduced features. Even by reducing the number of features, using geodesic distance did not improve the performance of the classifiers significantly.

3.2.10. Model selection.

The table below illustrates the final performance measures for the best performing models across all the previous chapters for a final evaluation.

Table 36. Final Model comparison

	Precision (Average)	Recall (Average)	F1 Score (Average)	Accuracy
Decision Tree (19 features)	0.83	0.83	0.83	0.83
Decision Tree (7 features)	0.83	0.83	0.83	0.83
Decision Tree (Geodesic-19 features)	0.83	0.82	0.82	0.82
Decision Tree (Geodesic – 7 features)	0.82	0.81	0.81	0.82
Random Forest (19 features)	**0.92**	**0.91**	**0.91**	**0.92**
Random Forest (7 features)	**0.94**	**0.93**	**0.93**	**0.93**
Random Forest (Geodesic-19 features)	0.86	0.86	0.85	0.86
Random Forest (Geodesic – 7 features)	0.87	0.87	0.87	0.88
XGBoost (19 features)	**0.93**	**0.93**	**0.93**	**0.93**
XGBoost (7 features)	**0.94**	**0.94**	**0.94**	**0.94**

XGBoost (Geodesic-19 features)	0.88	0.87	0.87	0.88
XGBoost (Geodesic – 7 features)	0.87	0.87	0.87	0.87
Stacked (19 features)	0.89	0.89	0.89	0.90
Stacked (7 features)	0.91	0.92	0.91	0.92
Stacked (Geodesic-19 features)	0.86	0.86	0.86	0.86
Stacked (Geodesic – 7 features)	0.84	0.83	0.83	0.84

Decision Tree Models:

- For Decision Tree models with both 19 features and 7 features, there is consistency in performance across precision, recall, F1 score, and accuracy, all achieving a score of around 0.83. This suggests that the model's predictive ability remains stable even with a reduced feature set. Though the models are prone to overfitting since there is a huge gap between training and testing performances.
- Introducing geodesic distance has a slightly different impact. While precision remains similar, there is a slight decrease in recall, F1 score, and accuracy. This may indicate that the geodesic distance introduces some complexity making difficult for trees to classify effectively and hindering predictive performance.

Random Forest Models:

- Random Forest models consistently outperform Decision Trees. The model with 19 features achieves high precision with 0.92, recall of 0.91, F1 score of 0.91 and accuracy of 0.92. By reducing the features, the performance of the model increases, achieving high precision of 0.94, recall of 0.93, F1 score of 0.93 and an increased overall accuracy of 0.93.

- Like Decision Trees, introducing geodesic distance in Random Forest models leads to a decrease in performance. The model with 19 geodesic features shows a noticeable drop in precision, recall, F1 score, and accuracy.

XGBoost Models:

- XGBoost models exhibit strong performance across all metrics, with the model using 7 features achieving high scores (precision = 0.94, recall=0.94, F1 score=0.94, accuracy=0.94). This highlights the effectiveness of XGBoost for the specific dataset.
- Like Decision Trees and Random Forests, introducing geodesic distance has a marginal negative impact on performance.

Stacked Models:

- Stacked models, which combine predictions from multiple base models, demonstrate competitive performance. The model with 7 features achieves high precision of 0.91, recall of 0.92, F1 score of 0.91, and accuracy of 0.92. However, they fall behind compared to the individual Random Forest and XGBoost models.
- Geodesic Distance again, decreased the performance for all metrics in the models.

Overall Observations:

- Random Forest and XGBoost consistently outperform Decision Trees and Stacked models across all metrics.
- The impact of geodesic distance varies across models, with a general trend of a slight decrease in performance. This suggests that while geodesic distance is easier to compute using the geopy library, there is an increase in complexity that hinders predictive performance in these specific scenarios.
- The choice of the number of features significantly influences model performance. Models with fewer features (7 features) often achieve higher precision, recall, F1 score, and accuracy compared to models with a larger number of features (19 features).
- The best performing model is XGBoost utilizing distance and 7 features in total.

3.2.11. Model Explainer

A fundamental challenge in machine learning models lies in comprehending feature importance and interpreting the outcomes. Although tree-based models visualize this importance through feature importance functions, these essentially reflect how the models, and thus individual trees, were constructed. Referring to the feature importance of Random Forest models, it is evident that distance and time consistently hold the highest positions in the list of influential features, irrespective of the number of features considered. This is illustrated in the plots below.

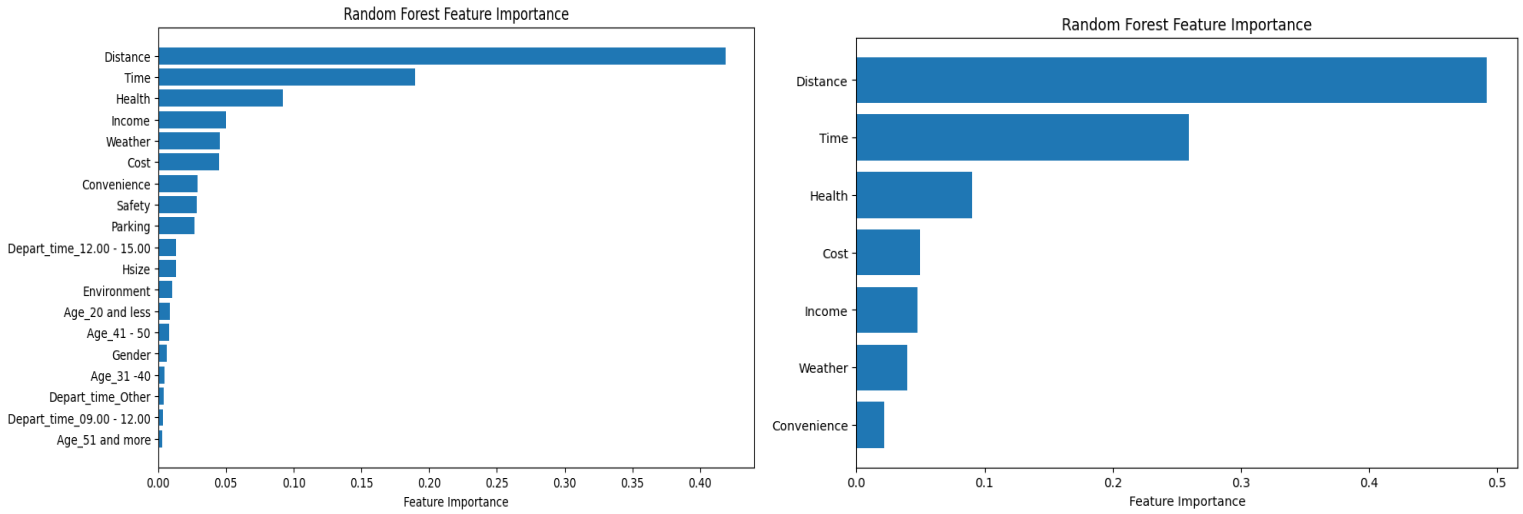


Figure 96. RF Feature Importance (7 vs 19 features)

However, the outcomes become less straightforward with the XGBoost model. While distance maintains its prominence as the most crucial feature, it appears that most of the features contribute significantly to the individual decision tree construction. This is particularly evident in the model employing seven features, where factors like physical activity and health take precedence over time in terms of importance. The subsequent features also demonstrate notable contributions. These findings are visually represented below.

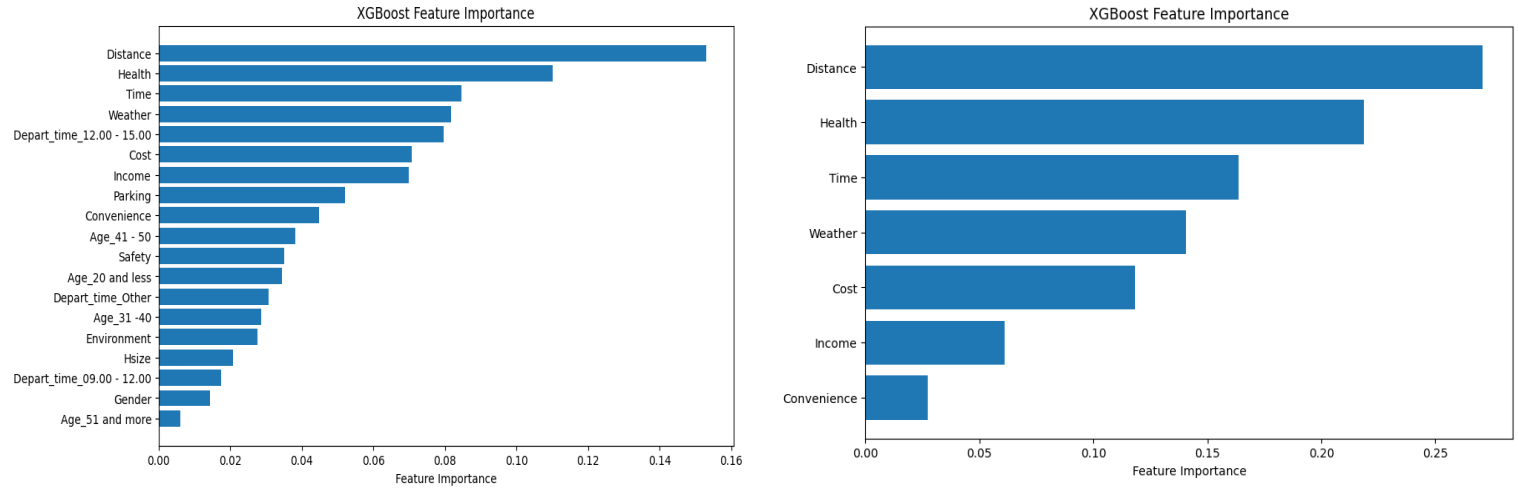
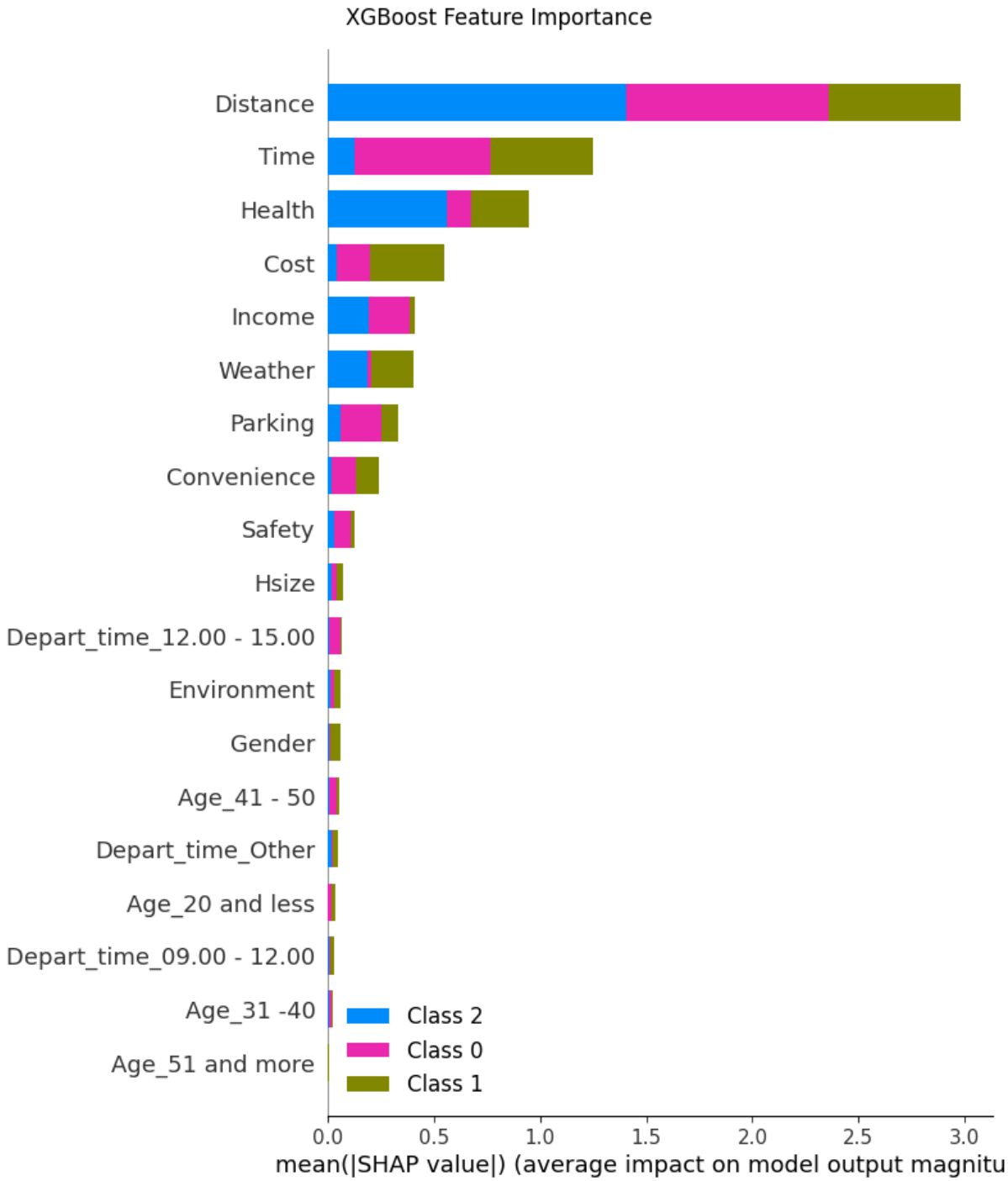


Figure 97. XGBoost feature importance (7 vs 19 features)

Nevertheless, a crucial question emerges: what feature truly holds the highest significance in guiding the model's prediction process, influencing the factors the model relies on for accurate predictions?
To address this question, the shap Python library comes into play. The library is constructed based on the "SHAP (Shapley Additive exPlanation) theory," as proposed by Lundberg & Lee (2017). SHAP is a procedure designed for interpreting the predictions of the models. It introduces a framework to explain a model's predictions by attributing the prediction to each individual feature. The SHAP values that the library computes, provide insights into how each feature influences individual predictions, highlighting the significance of each feature relative to others and showcasing the model's dependence on feature interactions. They offer a consistent and objective method for explaining the feature contribution to the model. Implementing a cooperative game theory approach, SHAP assigns weights for each feature. Assume a cooperative game with players equal to the number of features. In this scenario, SHAP reveals the individual contribution of each "player" (or feature in machine learning) to the model's output for every example or observation (Trevisan, 2022). Features with positive values exert a positive contribution on the predictions, while negative ones exert a negative contribution. The magnitude of those values demonstrates the strength of that influence (Awan, 2023).
The shap explainer function was used to construct the feature importance for XGBoost model (19 features). The code is depicted below. Note that ‘best_model’ refers to the tuned model with the best parameters found through grid search.

```
explainer = shap.Explainer(best_model)
shap_values = explainer.shap_values(X_test)
```

A plot was then created to display feature importance in summary. The plot displays the mean SHAP values of each feature and for each class label.



Class 0: Private Vehicle, Class 1: Bus, Class 2: Walk

Figure 98. XGB SHAP importance – 19 features

From the plot it is identified that Distance was indeed the most important feature for predictions regardless of class label. Time was the second most important feature, while the magnitude of the importance was higher for private vehicle – bus and lower for walk label. In contrast, the magnitude of importance is higher for walk label and Health which was the third most important feature for predictions. Below convenience, it is evident that additional features do not contribute that much.

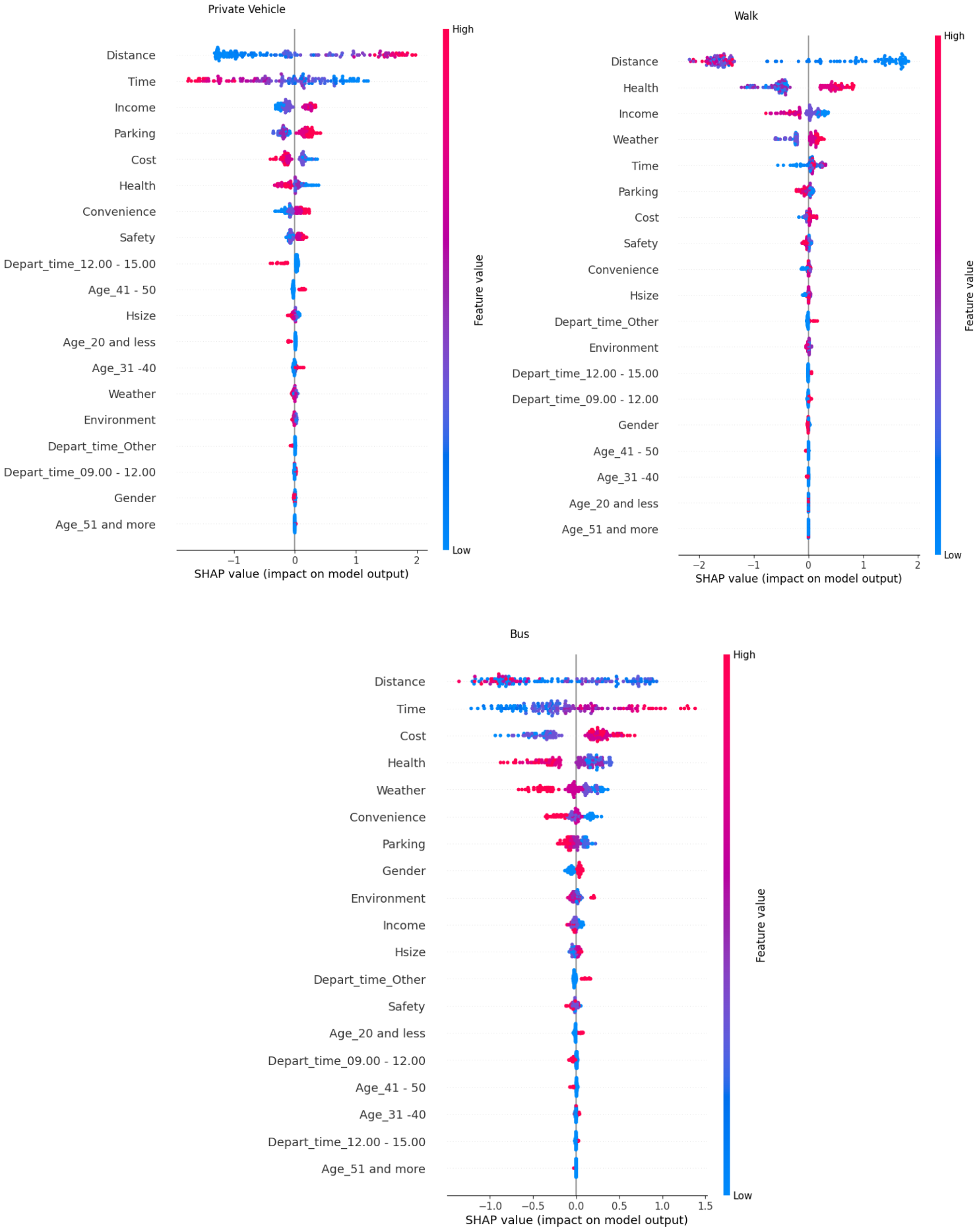


Figure 99. Class - wise feature importance

The above plots illustrate the impact of each feature for each class on the whole dataset. Each scatter represents an observation of the test set. Each colour describes the feature value (red for high and blue for low) while the values of the X axis indicate the impact on the outcome. The features are put in descending order from the most important feature to the least important one. A plot is generated for each class respectively. To interpret the plot, focus should be given at a specific class. For instance, by observing the plot of Walk class, it is identified that distance was the most important feature, followed by health. Low values (blue colour) for Distance (referring to short travel distances in kilometres) have a high positive contribution for predicting that class. Additionally, high values (red colour) for Health (referring to Agree and Strongly Agree on the impact of physical activities and health) have also a positive contribution on predicting that class, though the magnitude is lower that of distance. In contrast, high values (red colour) for Income (referring to commuters with income above 1250) have a negative contribution on predicting that class. In a multiclassification problem the negative contribution implies not predicted as that class (Walk vs rest of modes). For a more generic interpretation, the shorter the distances for commuting to work, the higher the tendency of the model to predict walking. In contrast, the highest the income, the lower the tendency of the model to predict walk.

Note that while Health is the second most important feature for predicting walk class, this is not the same for private vehicle and bus modes. By observing the plot for Bus, Distance is first followed by Time. The interpretation is the same as before. Higher values (red colour) for Time (referring to the minutes required for commuting to work) have a high positive impact in predicting that class. The generic interpretation is still, the higher the commute time to work, the higher the tendency of predicting bus. In contrast, higher values (red colour) for Health (Agree, strongly Agree), have a negative impact in predicting that class. Hence, as the ranking position increases, from agree to strongly agree, the tendency of predicting bus decreases (higher chance that the commute happens with other mode and possibly walking). The same pattern could be followed for every feature and for every class in trying to explain the outcome of the prediction. The same graphs were computed for XGBoost model utilizing 7 features. The summary plot of feature importance is illustrated below.

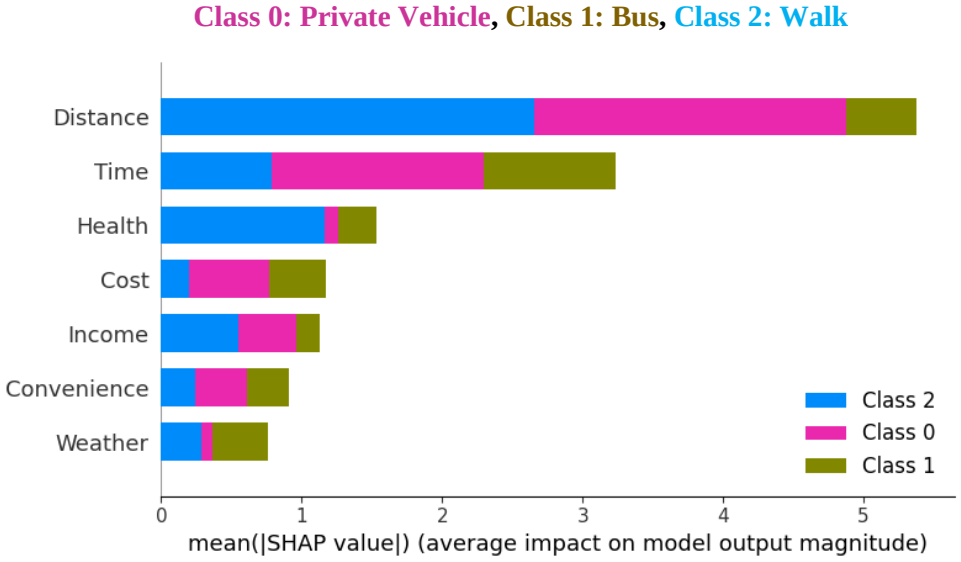


Figure 100. XGB SHAP importance - 7 features

Once more, Distance emerges as the paramount feature but only for private vehicle and walk now, compared to all classes in the previous model. For bus, Time has surpassed Distance as the most crucial feature, while Health is the second most important for walk labels, as indicated by the magnitude of the values. The three individual plots were constructed for a better understanding of the feature importance on predictions.

Figure 101. Class - wise importance

By observing the plot for Bus, the interpretation is the same as before. Higher values (red colour) for Time (referring to the minutes required for commuting to work) have a high positive impact in predicting that class. The generic interpretation is still, the higher the commute time to work, the higher the tendency of predicting bus. In contrast, for private vehicle, the higher the distance, the higher the tendency of predicting private vehicle. The same procedure can be followed for every iteration of class and features just by observing the individual plots.

In general, SHAP analysis offers a more detailed and individualized approach to feature importance compared to traditional methods like retrieving feature importance for a tree-based model in scikit-learn. While scikit-learn provides feature importance based on metrics like Gini impurity or information gain, it tends to offer a global ranking without considering the interaction effects between features. SHAP, on the other hand, provides a specialised approach that considers the contribution of each feature for every individual prediction. This allows for a better understanding of how each feature influences specific instances, capturing both main effects and interactions. Shap library contains additional plots that could be investigated, such as dependency plots, while it is possible to explore the magnitude of each prediction of the model. Although this kind of analysis is not explored is this Thesis, it could very well be a fundamental context for research in another work.

4. Case 2 – Netherlands

4.1. Methodology

4.1.1 Data collection

The data for the Netherlands were collected from the statistics service in the country. The data can be found at **https://ssh.datastations.nl/dataverse/root**. For data to become available to the user, registration to the website is required, while access to the datasets must be requested by the "Data station of Social Sciences and Humanities" for free. Data was collected for years 2018 - 2022. The datasets are trip diary surveys where people within the Netherlands record their trips and answer demographic questions. The survey is called "ODIN" and is conducted by the "Central Bureau of Statistics" in the Netherlands annually via the Internet (Statistics Netherlands, 2024). Features within the dataset are described in Dutch so it was required to translate them using Google Translate and following the Codebook Guideline also available at the same link as above. The original datasets for each year, as downloaded from the website, contain about 150000-200000 trips/rows and about 100 columns. From these datasets, trip related features were selected such as distance and duration of the trip and some demographic features such as household size and number of vehicles in the house. After the features were extracted for each year, the data were merged and a final dataset was produced. The resulting dataset contains 817126 samples, each instance representing a different trip. After removing rows with blank values, the dataset was imported into python for further processing. The features are illustrated below:

```
Index(['HHPers', 'Geslacht', 'Leeftijd', 'Herkomst', 'Opleiding',
       'OPRijbewijsAu', 'HHAuto', 'HMopeds', 'HHEFiets', 'Jaar', 'Maand',
       'Weekdag', 'Feestdag', 'Toer', 'KMotiefV', 'AfstR', 'RReisduur',
       'RVertUur', 'KRvm'],
      dtype='object')
```

The features were then renamed so they could become clearer and more manageable. The new features names are illustrated below.

```
Index(['People_in_house', 'Gender', 'Age', 'Background', 'Education',
       'Driver_license', 'Cars_in_house', 'Mopeds_in_house', 'Electric_bike',
       'Year', 'Month', 'Weekday', 'Holiday', 'Round_trip', 'Motive',
       'Distance', 'Duration', 'Hour', 'Mode'],
      dtype='object')
```

The resulting dataset comprises 817126 samples and 20 features. The analysis procedure that was followed for the dataset is illustrated in the figure below:

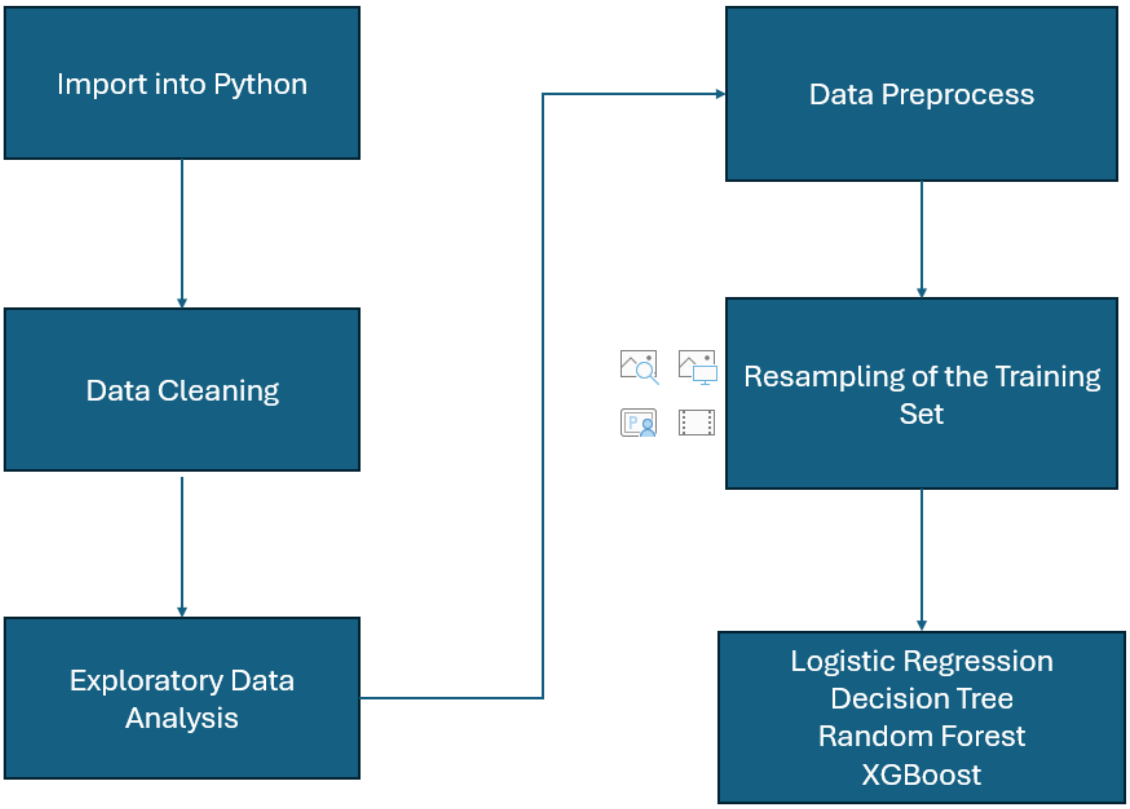


Figure 102. Case 2 - Analysis Procedure

The target variable (transport mode) is imbalanced. The percentages of each mode are:

Car: 46.63 %

Bike: 25.16 %

Walk: 24.51 %

Public Transport: 3.70 %

Such class imbalance could create bias to the classifiers. To address this situation resampling strategies were implemented. More specifically, random undersampling and random oversampling were implemented on the training set. The undersampling strategy removes samples from the majority classes so to much the minority. In contrast, the oversampling strategy duplicates samples from the minority classes so that they match the majority class. This thesis experiments with both resampling strategies and compares the performance of the classifiers implementing each strategy. Note that both resampling strategies were implemented strictly for the train set while the test set remained unbalanced to better represent real-world problems associated with transport modes. The resampling strategies were combined with python pipelines from the imblearn library package. The pipeline is used to chain many processing steps into a single estimator. Hence, normalization scaler, resamplers and classification models were all combined into the pipeline to automate the training process and evaluate the models on the testing set.

Four models were considered for the problem: Logistic Regression, Decision Tree, Random Forest and XGBoost. Considering the imbalance problem, precision, recall and f1-macro metrics were primarily evaluated for model performance. Additionally, the precision – recall curves were constructed to evaluate the area under the curve, called Average Precision (AP). Better AP scores indicate a better performance for the model. Since this is a multiclassification problem, curves are constructed for each individual class in a OneVSRest scenario. The Micro-AP was alco computed representing the weighted average of each individual class AP.

The model performances were also compared to the corresponding models proposed at the paper by Kashifi et al (2022). The authors also investigated transport patterns and classification of mode within Netherlands implementing both undersampling and oversampling strategies. They used travel diary data originating from the same source (National Bureau of Statistics). The distinction with their research lies within the data themselves. The authors used travel diary data from 2010 – 2012. Their dataset comprised of 230.608 trips and 17 features, including population density and weather condition metrics. This Thesis differs by experimenting with more recent data (2018 to 2022), while the size of the dataset is significantly larger. Also, there is a differentiation in features used, encompassing elements such as trip duration, trip motive, departure hour, and date stamps like the weekday and month occurrences of the trip. Conducting experiments with diverse sets of features and updated data is crucial for a more comprehensive grasp of mobility patterns and to identify potential variations over time as countries and transport infrastructures constantly evolve.

4.2. Results for Netherlands

In this section of the Thesis, the results from the Netherlands trip diary will be presented. The initial part covers data preparation and cleaning, followed by exploratory data analysis and data preprocessing. The final segment focuses on model application, mode classification, and the selection of the best model.

4.2.1. Data preparation and cleaning

The values for each feature in the entire dataset were in numerical form so it was essential to encode them back to categorical. The procedure was followed using the Codebook guideline at: https://ssh.datastations.nl/dataset.xhtml?persistentId=doi:10.17026/dans-zwz-fq8t .

After categorical features were encoded, the dataset was checked for null values and duplicates to drop. The dataset contained 37213 duplicated rows, so they were removed. Below is the list of features used.

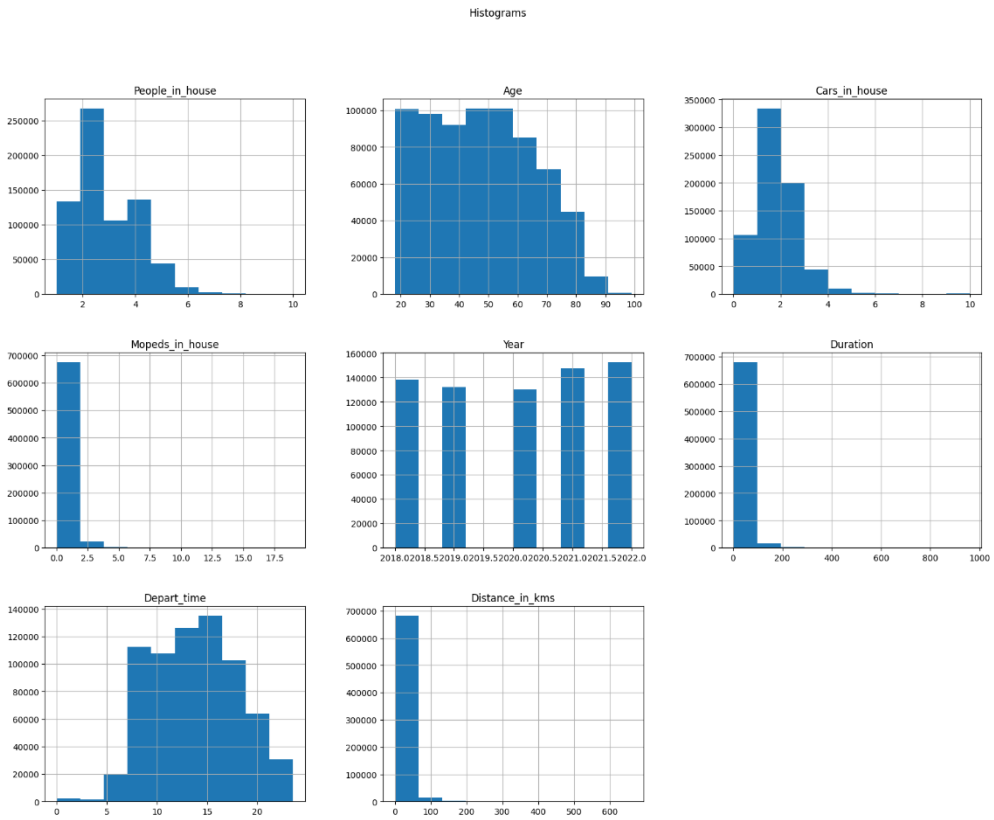
1. People_in_house - How many members the household has - Numeric - Discrete - Integer
2. Household_composition - eg Parents + child or Single parents - Categorical - Nominal
3. Gender - Male / Female - Categorical - Binary
4. Age - Age of the respondent in years - Numeric - Continuous - Integer
5. Background - The citizenship of the respondent (Dutch or other) - Categorical - Nominal
6. Education - Highest education completed for the respondent - Categorical - Nominal
7. Drivel_License - If the respondent has car driving license - Categorical - Binary
8. Cars_in_house - How many vehicles the household has - Numeric - Discrete - integer
9. Mopeds_in_house - How many mopeds(type of motorcycle) the household has - Numeric - Discrete - Integer
10. Electric_bike - If the respondent has access to an electric bike - Categorical - binary
11. Year - Year that the trip occurred - Numeric - Integer
12. Month - Month that the trip occurred - Categorical
13. Weekday - Day that the trip occurred - Categorical
14. Holiday - If the trip occurred on a national holiday - Categorical - Binary
15. Round_trip - If the respondent made a round trip (Begin and finish the trip at the same location) - Categorical - Binary
16. Motive - The reason of the trip - Categorial - Nominal
17. Distance - The distance (In hectometres) of the trip - Numeric - Continuous
18. Duration - The duration (In minutes) of the trip - Numeric - Continuous
19. Hour - Hour of departure
20. Mode - The mode of transportation for the trip - Categorical - Target variable

Year was dropped from the dataset since it was not relevant for the analysis. The table below illustrates descriptives for some of the features.

Distance	Duration	Hour
779309.000000	779309.000000	779309.000000
97.432447	24.076450	13.629871
199.654405	31.751656	4.254536
0.000000	0.000000	0.000000
11.000000	10.000000	10.000000
30.000000	15.000000	14.000000
86.000000	30.000000	17.000000
6600.000000	993.000000	33.000000

For the "Hour" feature, indicating the departure hour of the trip, a maximum value of 33 was identified as erroneous. Subsequently, samples with a departure hour exceeding 23 were removed. Additionally, the Distance column measures the trip distance in hectometres. Therefore, the values were transformed in kilometres for better handling. Furthermore, there were trips where duration and distance were 0 respectively. Those samples were also removed from the dataset. Finally, only trip records of respondents above 17 years old were considered.

The following picture illustrates the histograms for all the numeric variables.



Observing both at the descriptives and the histograms, it is identified that for Distance, Duration and Mopeds there is a presence of outliers. Therefore, samples were dropped when the Distance was above 200 kilometres. Samples were also removed when the duration of the trip was above 300 minutes. Last, samples were removed when the respondent of the trip had more than 6 mopeds in the house. The final dataset, after processing and cleaning, consists of **663745** samples**.**

4.2.2. Exploratory Data Analysis

In this section of the Thesis the descriptive statistics for this case, along with visualisation for each feature will be presented to better understand the dataset. The first feature is about the target variable containing the transportation mode of the trip. Most trips were undertaken using a car (46.63%), either as a passenger or driver, followed by those made on a bike or on foot. Public transportation trips were significantly lower compared to other means, accounting for only 3.56%.

Mode	
car	**46.63%**
bike	**25.16%**
walk	**24.51%**
public transport	**3.69%**

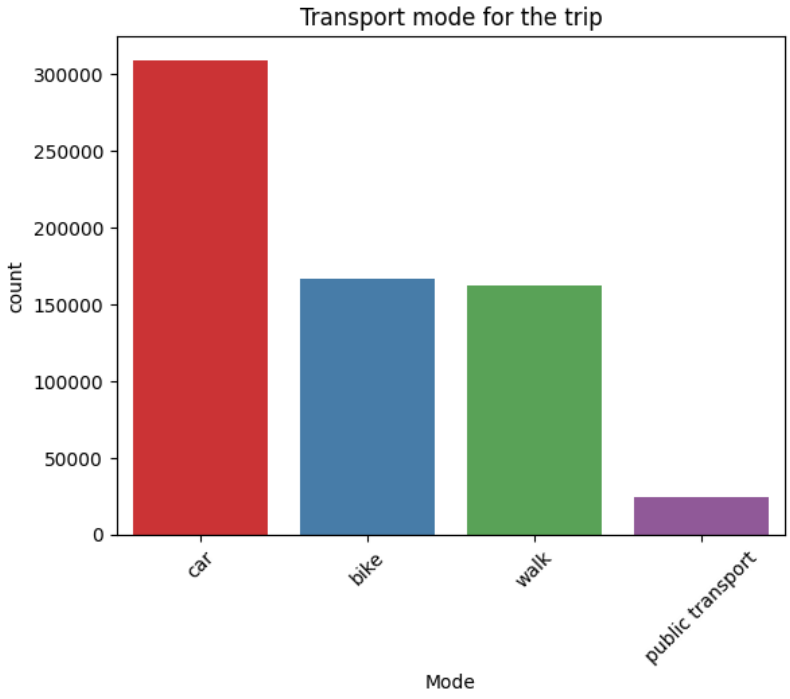


Figure 103. Transport Mode

The next feature is about the month that the trip occurred. The outcome exhibits a balanced pattern, indicating that the trip count is consistently above 50,000 observations for each month. September stands out with the highest number of trips, representing 9.18%. In contrast, April recorded fewer trips, constituting 7.5% of the observations.

Month	
September	9.18%
November	9.00%
October	8.98%
January	8.93%
May	8.32%
March	8.30%
August	8.24%
June	8.05%
July	7.98%
December	7.86%
February	7.61%
April	7.50%

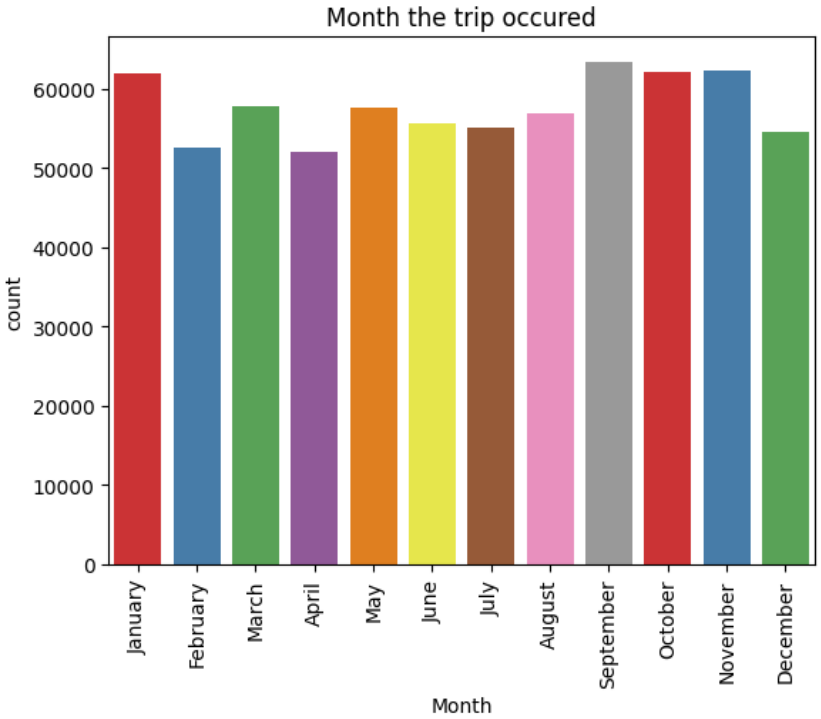


Figure 104. Month

The following feature refers to the day of the week that the trip occurred, with the trip count exceeding 90,000 observations on almost every day. The sole exception lies in trips made on Sundays, totalling just under 80,000 observations, and representing 10.94%.

Weekday	
Friday	15.77%
Thursday	15.11%
Tuesday	14.73%
Saturday	14.58%
Wednesday	14.57%
Monday	14.27%
Sunday	10.94%

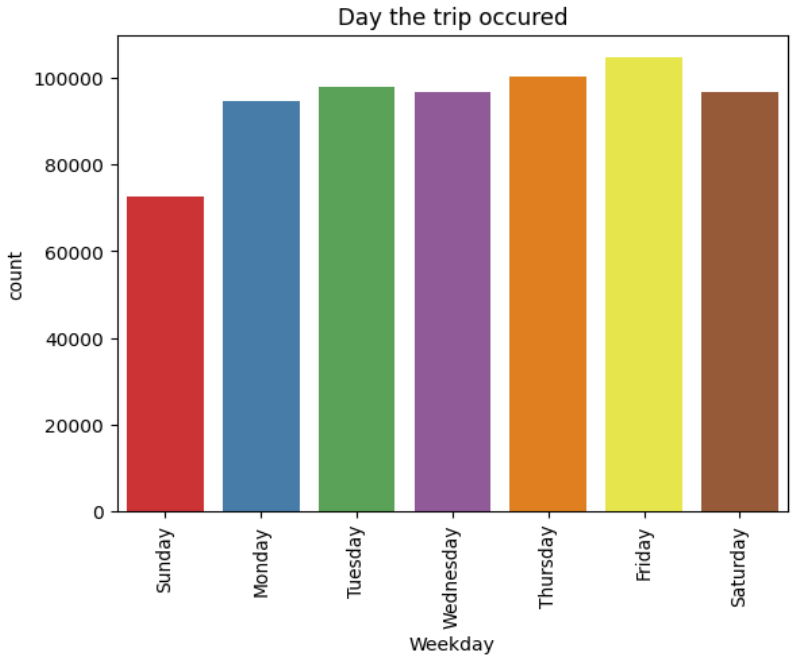


Figure 105. Weekday

The subsequent feature is about the gender of the respondent. Males account for 51.83 % of the total trips while females account for the remaining 48.16 %.

Gender

Female 51.83%

Male 48.16%

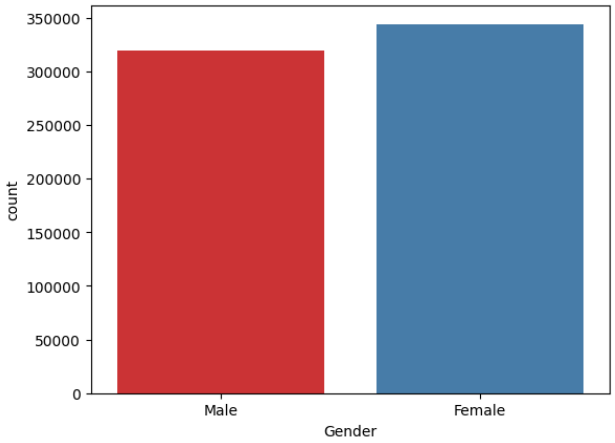


Figure 106. Gender

The following feature refers to whether the respondent of the trip has access to an electric bike. Most respondents account for "no" with 69.80%, while the remaining 30.20% indicates that they have access in some capacity.

Electric_bike

No 69.80%

Yes 30.20%

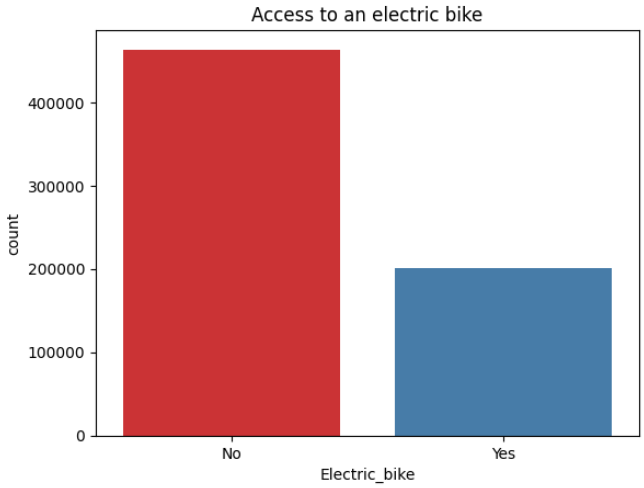


Figure 107. Access to electric bike

The next feature refers to the reason for the trip. Most trips were undertaken for shopping and grocery purposes, succeeded by trips to and from work. Additionally, a substantial proportion was attributed to social activities and journeys from/to work. The least frequent observations were associated with purposes related to education, business, and personal care, each totalling around 20,000 counts. Other motives also refer to visits, tours or hiking related.

Motive	
Shopping/Grocery	**21.31%**
Go to/return from work	**19.75%**
Social and recreational	**16.17%**
Other motives	**11.59%**
Tours/hiking	**11.36%**
Visits	**10.2%**
Service/Personal care	**3.54%**
Education	**3.19%**
Business related	**2.84%**

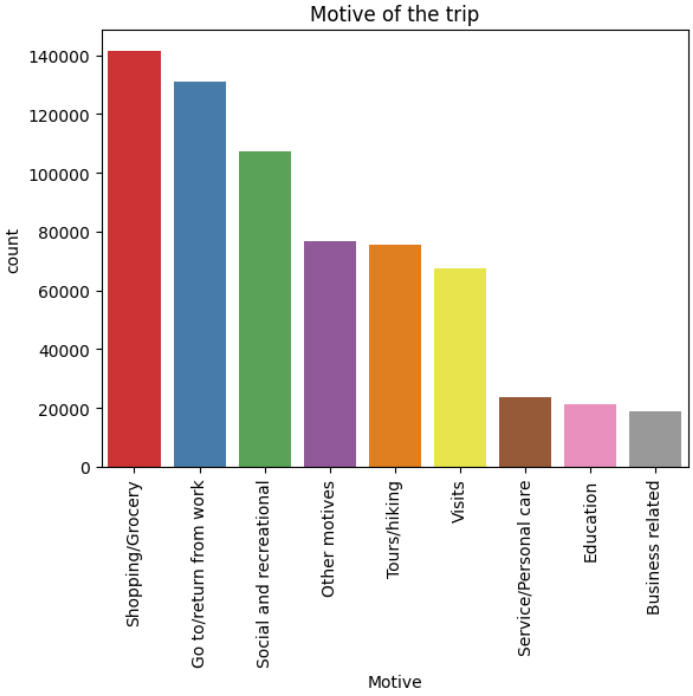


Figure 108. Motive of the trip

The subsequent feature implies whether the respondent made a round trip or not. A round trip refers to a scenario where the initial point is the same as the destination point. For example, this situation arises when the respondent commences the trip from home and concludes it at home as well. Such activities may likely involve physical exercise. The plot reveals that a minimal proportion of trips were genuinely round, constituting 12.99%. Hence, most trips had distinct starting and ending locations.

Round_trip	
No	**87.01%**
Yes	**12.99%**

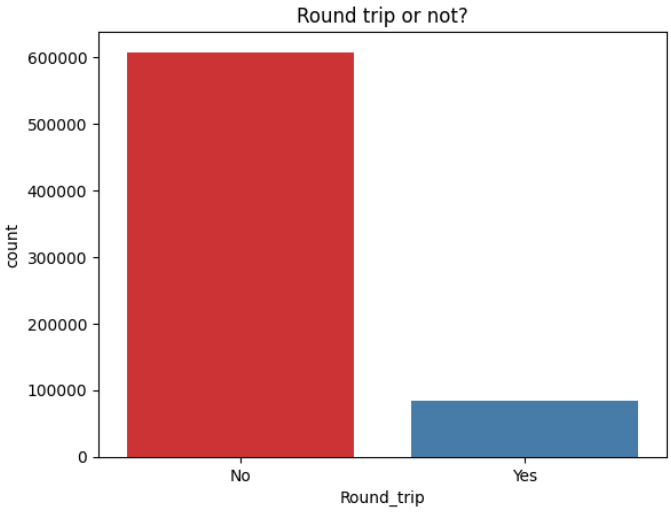


Figure 109. Round trip

The next feature refers to the number of mopeds that the respondent of the trip has in his household. The moped is considered a motorised bicycle offering a sustainable transportation option that often involves less stringent licensing requirements compared to conventional motorcycles and automobiles. (Wikipedia, 2024). The overwhelming majority of the trips were made by respondents that had 0 mopeds in their household (84.86%).

Mopeds_in_house

0 84.86%

1 11.65%

2 2.57%

3 0.60%

4 0.19%

5 0.07%

6 0.03%

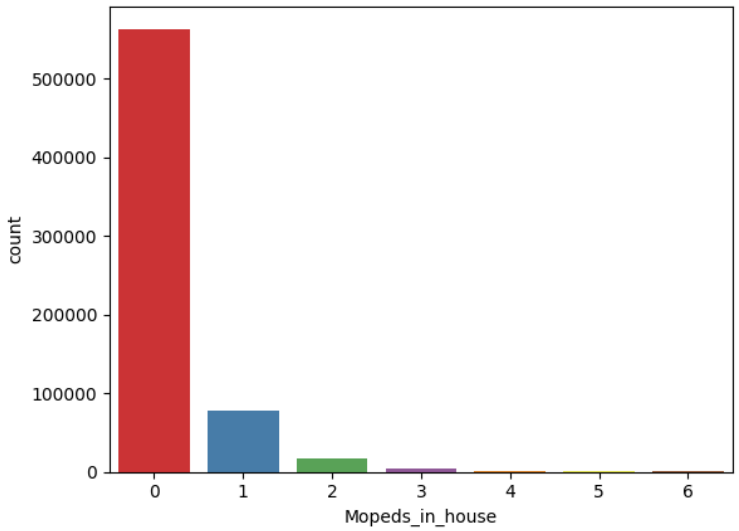


Figure 110. Number of mopeds

The following feature, same as mopeds, pertains to the count of cars in the household of the trip respondent. Most trips were undertaken by respondents with 1 or 2 cars in their homes. A substantial number of trips also involved respondents with zero cars in their households, while a smaller percentage comprised respondents with three or more cars.

Cars_in_house

1 47.63%

2 28.65%

0 15.14%

3 6.34%

4 1.49%

5 0.35%

10 0.17%

6 0.11%

7 0.04%

9 0.02%

8 0.02%

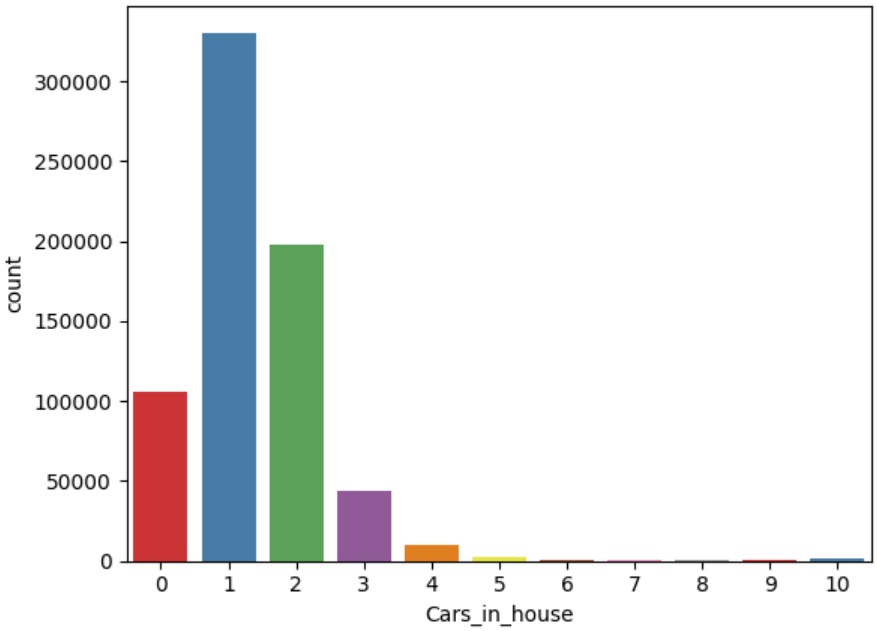


Figure 111. Number of cars

The next feature is about whether the trip was made by respondents with a driving licence or not. The biggest proportion of trips was conducted by respondents with an active car driving licence (87.33 %).

Driver_license

Yes 87.33%

No 12.67%

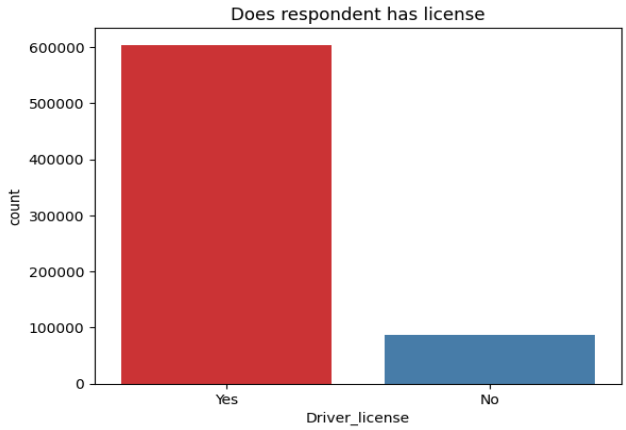


Figure 112. Driver license

The next feature pertains to the count of people in the households of the respondents who undertook the trip. The graphical representation indicates that most of the trips were carried out by respondents with 2 household members (38.23%), including themselves. Additionally, a significant proportion reported being part of a single-member family (19.03%). The proportion diminishes for trips made by respondents with 5 or more family members in their homes.

People_in_house

2 38.23%

4 19.31%

1 19.03%

3 15.23%

5 6.28%

6 1.40%

7 0.33%

8 0.09%

9 0.04%

10 0.02%

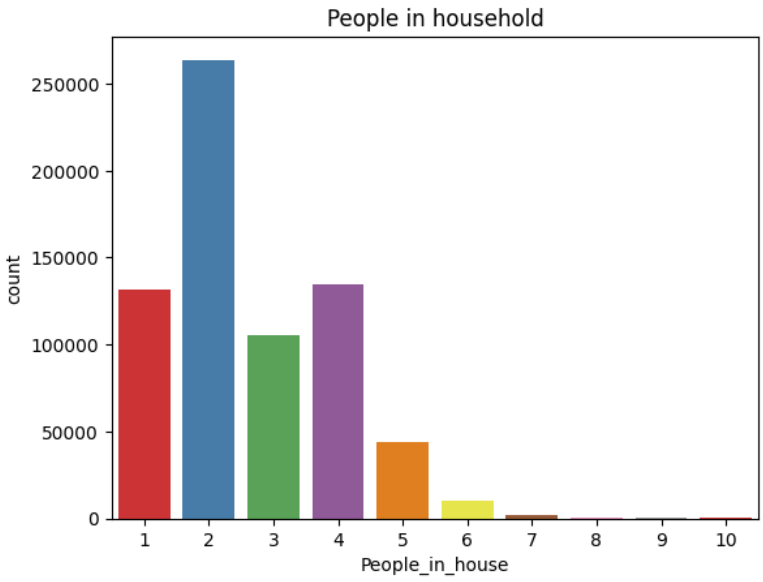


Figure 113. People in house

The subsequent feature pertains to the educational background of the respondent who conducted the trip. Most respondents have completed a higher education degree (48.51%), followed by those who completed secondary education (38.85%). A smaller proportion of respondents have completed other types of education, including vocational, primary, and other forms.

Education	
Higher education	**48.51%**
Secondary	**32.91%**
Lower vocational	**13.77%**
Other education	**2.34%**
Primary education	**1.79%**
No education	**0.66%**

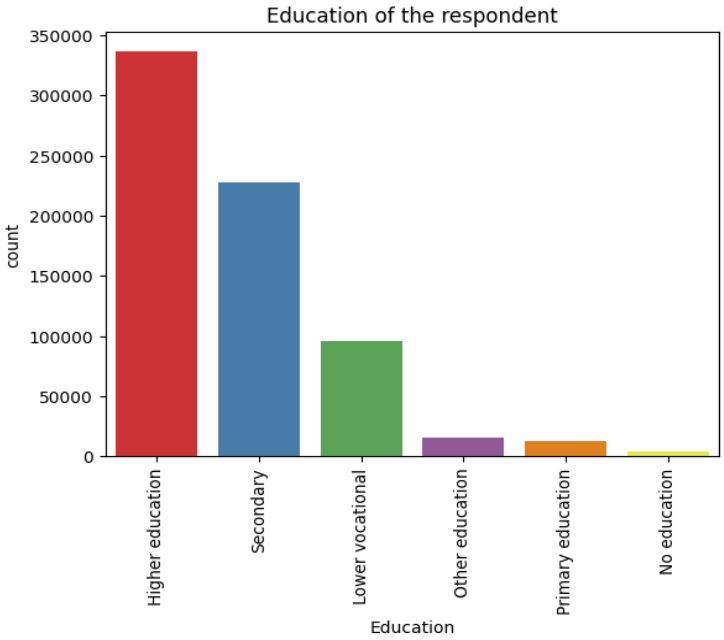


Figure 114. Education

The final categorical variable pertains to the background of the respondent. The highest proportion involves trips conducted by Dutch native residents (80.07%), while a lower proportion involves trips by non-Dutch respondents (19.92%).

Background	
Dutch	**80.07%**
Non Dutch	**19.92%**

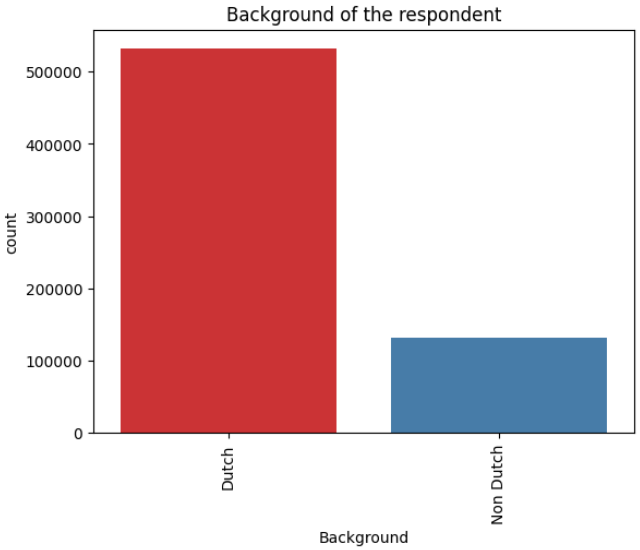


Figure 115. Background

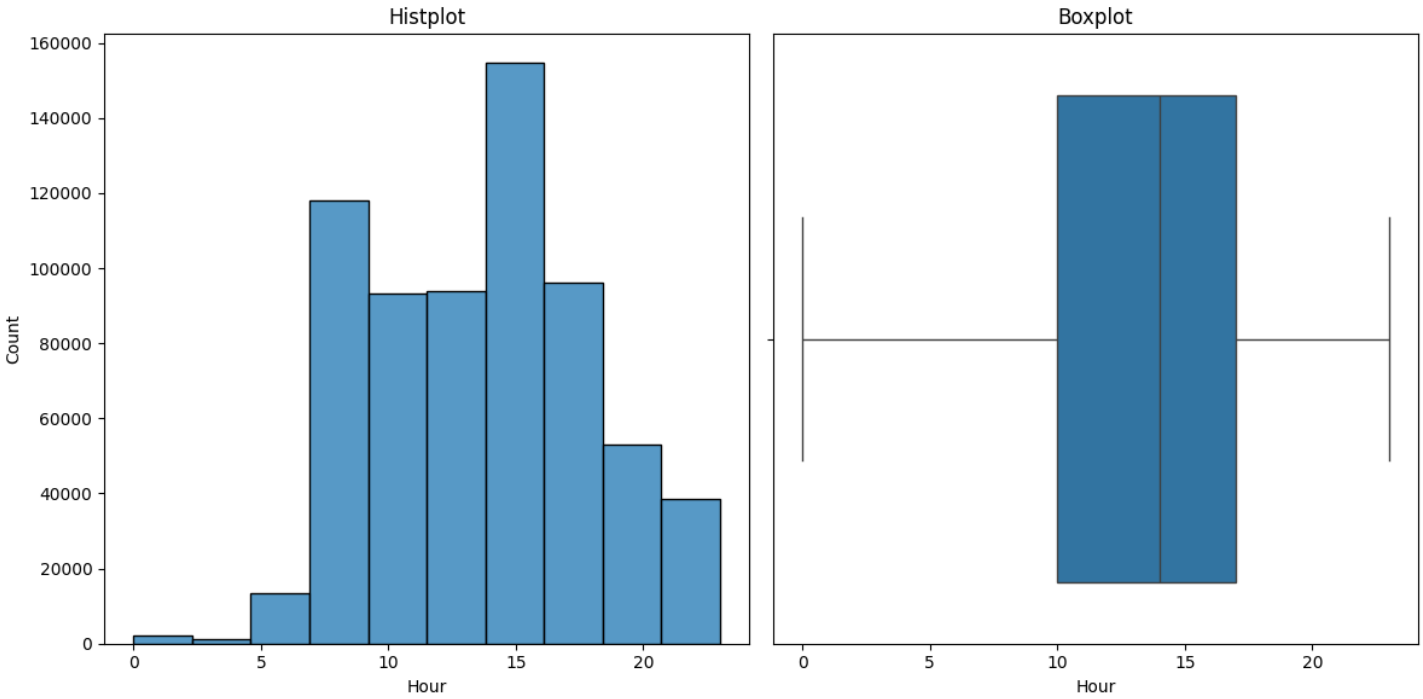


Figure 116. Hour of the trip

The figure above showcases the histogram and boxplot of the departure time. The mean departure time of 13.63 suggests a central tendency around the early afternoon, while the standard deviation of 4.29 indicates a moderate level of variability in departure times. The dataset's spread ranges from a minimum of 00:00 to a maximum of 23:00, illustrating a full day's span. Quartiles further characterize this distribution, with 25% of departures occurring by 10:00, 50% by 14:00 (the median), and 75% by 17:00

Table 37. Depart hour descriptives

	Departure time
Mean	**13.63**
std	**4.29**
min	**00.00**
25 %	**10.00**
50 %	**14.00**
75 %	**17.00**
max	**23.00**

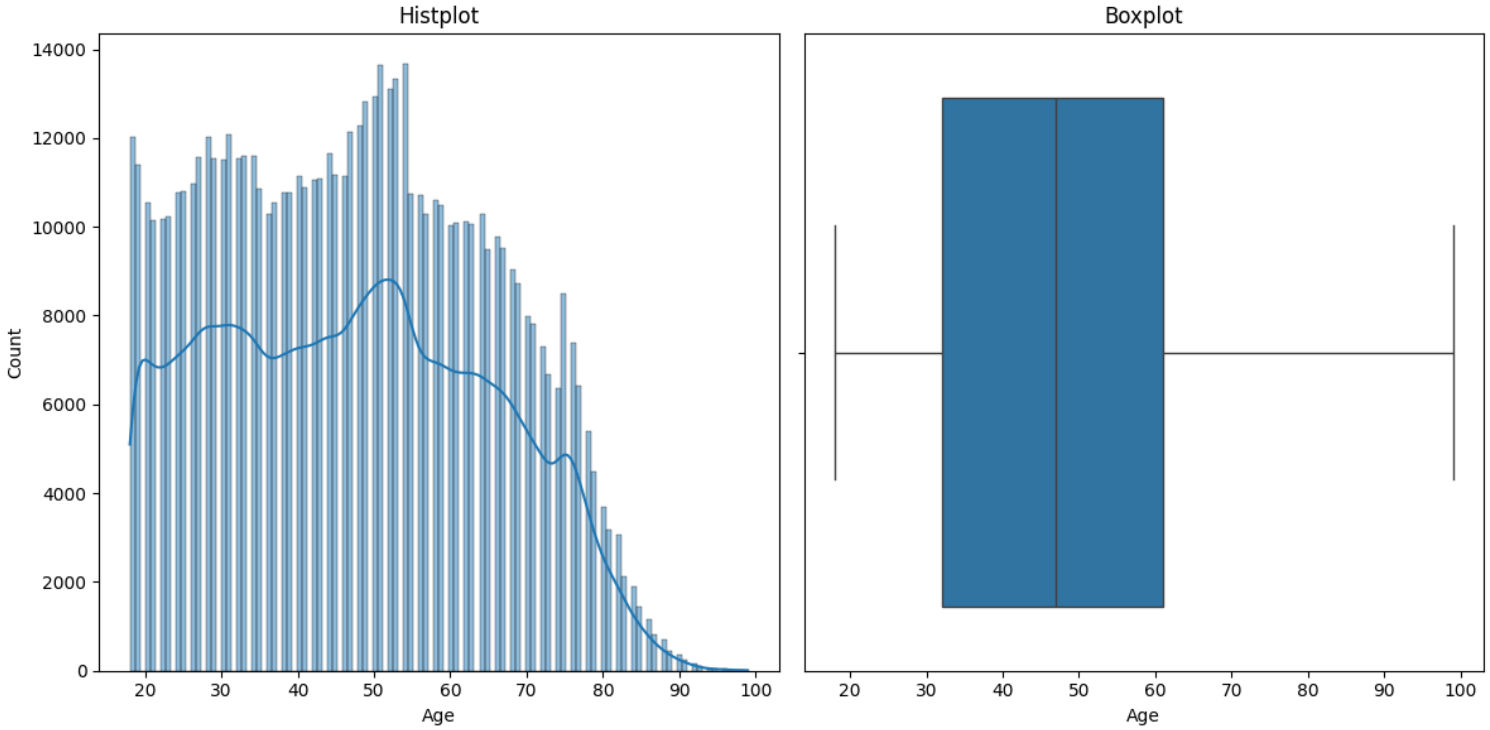


Figure 117. Age

The mean age of 47.50 indicates the central tendency, suggesting that the average age is around the late 40s. The standard deviation of 17.73 represents the variability or spread in ages. The dataset's age distribution spans from a minimum of 18 to a maximum of 99, revealing a diverse range of ages. The quartiles further characterize the distribution, indicating that 25% of individuals are aged 32 or younger, 50% are aged 47 or younger (the median), and 75% are aged 61 or younger.

Table 38. Age descriptives

	Age
Mean	**47.50**
std	**17.73**
min	**18**
25 %	**32**
50 %	**47**
75 %	**61**
max	**99**

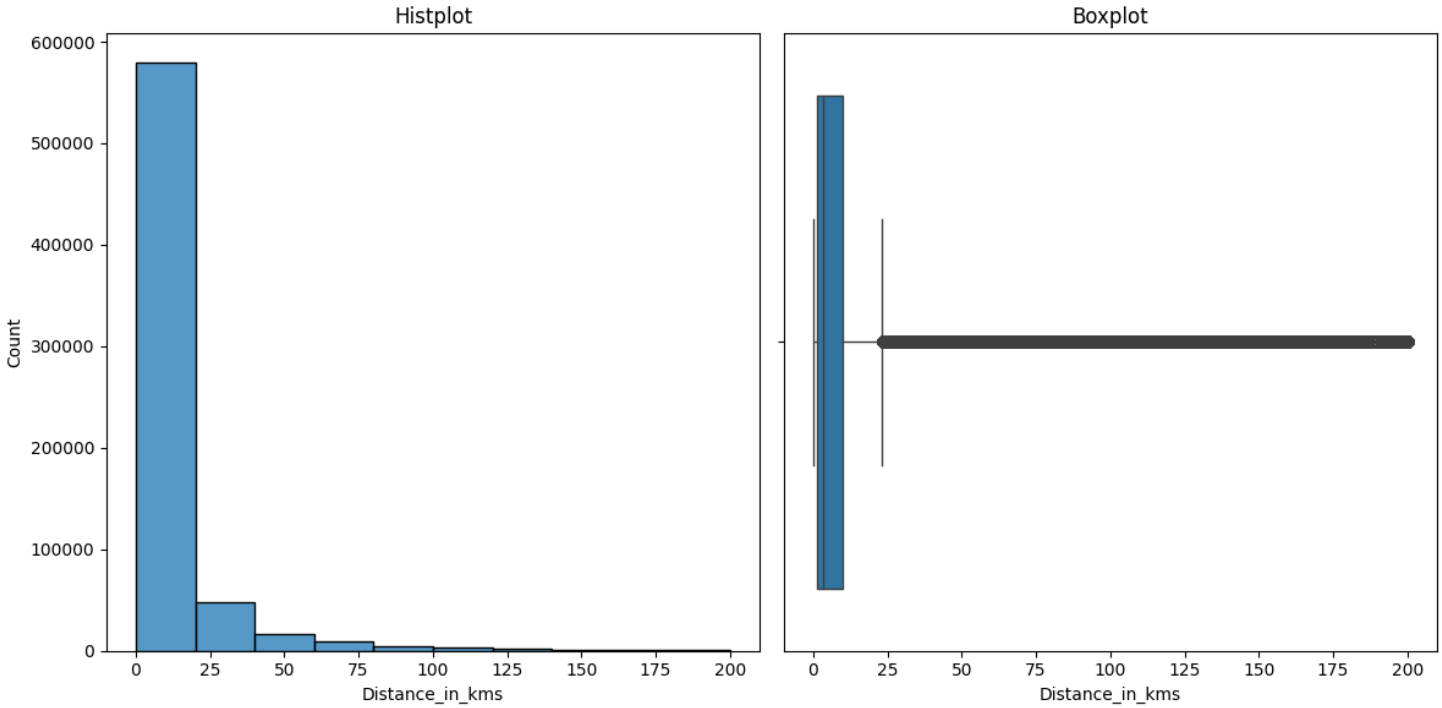


Figure 118. Distance in kilometres

The mean distance of 10.22 suggests a central tendency around the low double digits, indicating the average distance travelled. The standard deviation of 19.63 highlights a significant variability or spread in distances, with some values potentially distant from the mean. The dataset's distance distribution spans from a minimum of 0.1 km to a maximum of 200 km, showcasing a wide range of travel distances. Quartiles further detail the distribution, with 25% of distances being 1.2 km or shorter, 50% falling at 3.3 km or below (the median), and 75% being 10 km or shorter.

Table 39. Distance descriptives

	Distance_in_kms
Mean	**10.22**
std	**19.63**
min	**0.1**
25 %	**1.2**
50 %	**3.3**
75 %	**10**
max	**200**

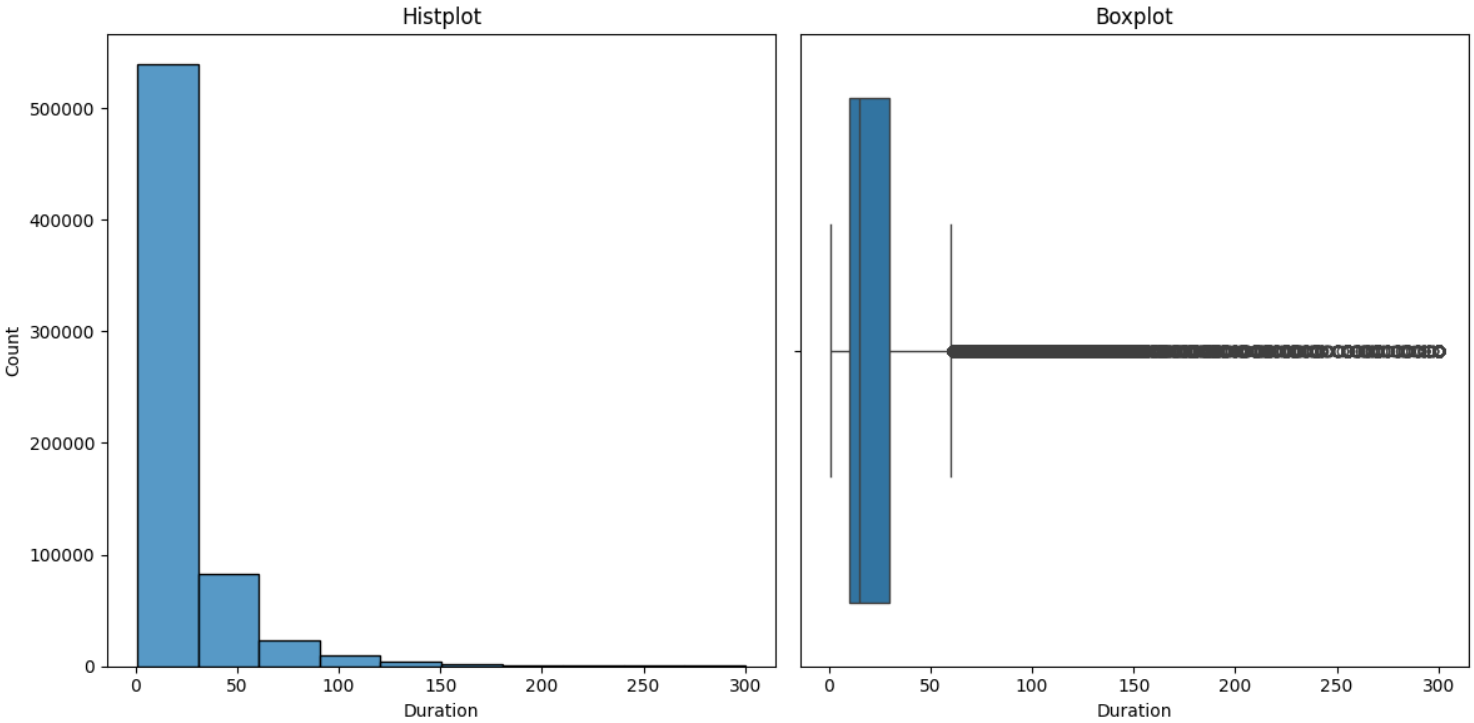


Figure 119. Duration

The mean duration of 24.03 suggests a central tendency around the mid-20s, indicating the average length of travel. The standard deviation of 28.1 reflects a notable variability or spread in durations, with some values potentially deviating significantly from the mean. The dataset's duration distribution spans from a minimum of 1 minute to a maximum of 300 minutes, showcasing a wide range of travel time. Quartiles further characterize the distribution, with 25% of durations being 10 minutes or shorter, 50% falling at 15 minutes or below (the median), and 75% being 30 minutes or shorter.

Table 40. Duration descriptives

	Duration (in minutes)
Mean	**24.03**
std	**28.1**
min	**1**
25 %	**10**
50 %	**15**
75 %	**30**
max	**300**

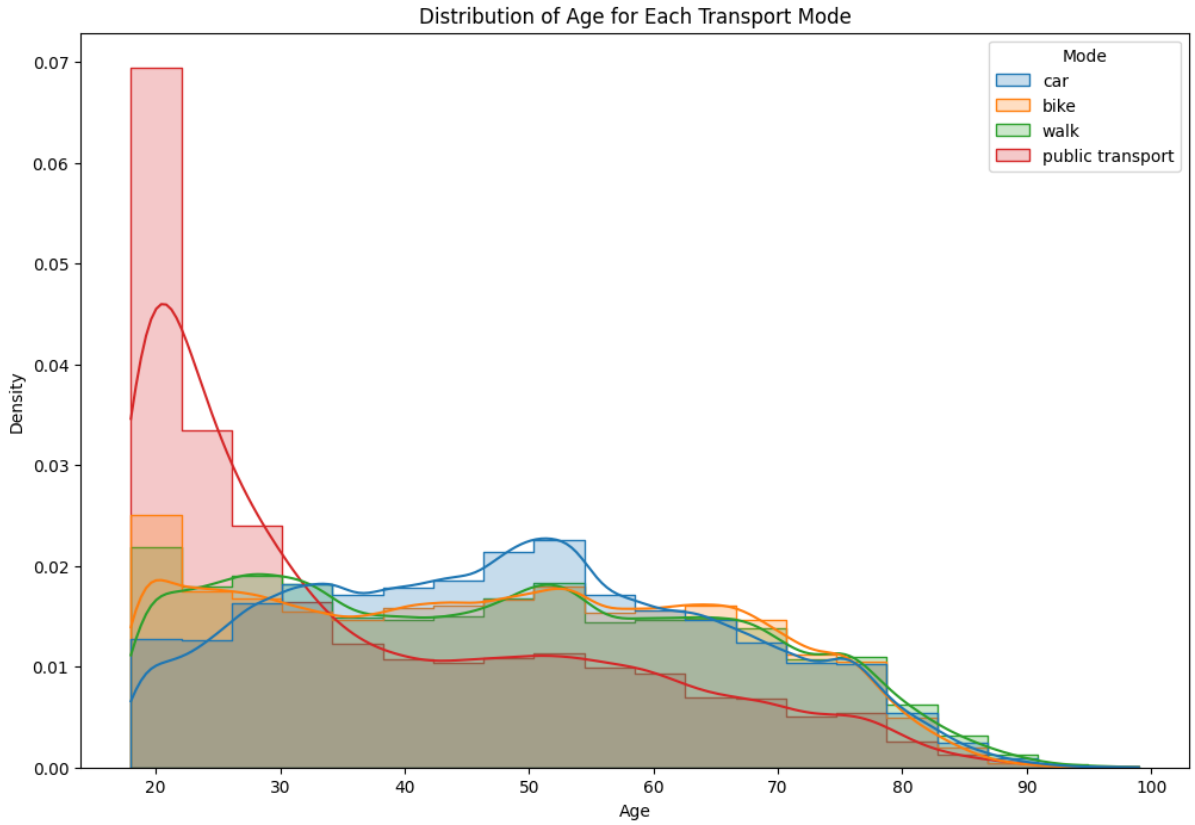


Figure 120. Age by mode

The plot above illustrates the age density for each transport mode. In the case of public transportation, most trips were undertaken by respondents aged between 18 and 30. The curves for other transport modes exhibit similar patterns, being smoother compared to public transport, indicating consistent age-related trends among trip respondents.

The plot below, illustrates the trip motives grouped by transport mode. A notable contrast arises when examining trip motives, particularly for tours or hiking, where walking overwhelmingly emerges as the preferred mode of transportation. In contrast, the minority mode, public transportation, mostly occurs in instances where the trip motive is commuting to or returning from work.

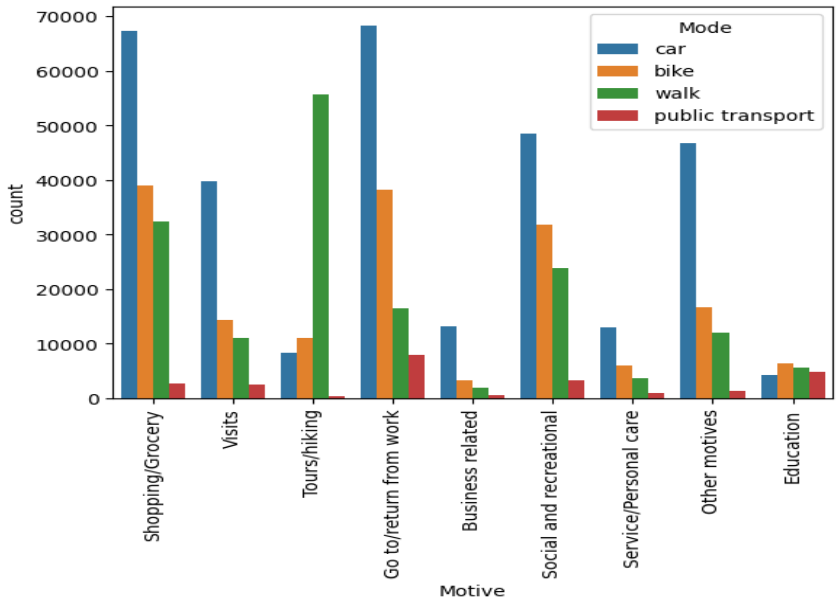


Figure 121. Trip Motive by mode

The figure below illustrates a density plot for the departure hour of the trip for each mode of transportation. Minimal variations are observed among the curves for car, bike, and walking options, indicating similarities to the departure hour patterns. In contrast, public transport stands out with two distinct peaks, suggesting that most trips using public transport occurred between 5 and 10 in the morning and between 15 and 17 in the evening. Notably, while the trends in curves for other modes align, the concentration of walks appears to be higher than other modes for trips occurring between 19 and 23 o'clock.

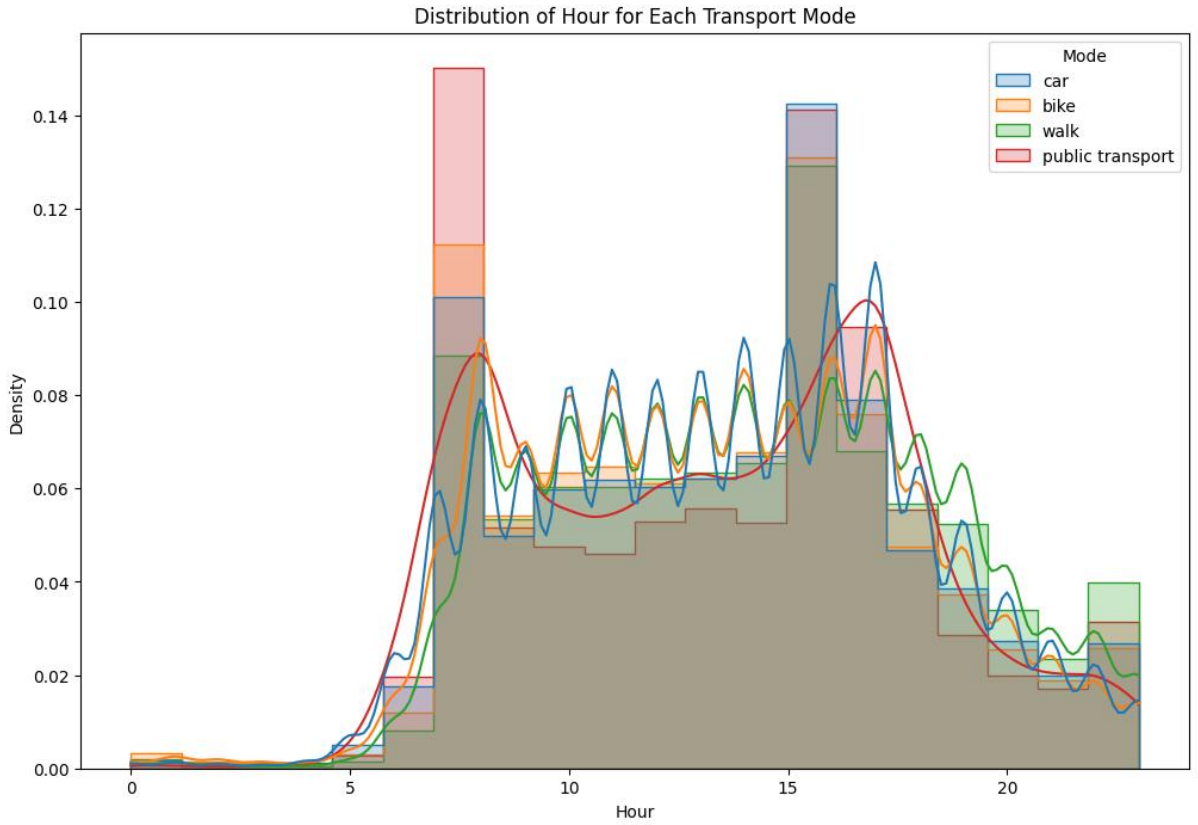


Figure 122. Density of depart hour for each mode

The subsequent figures depict the Month and Weekday features categorized by transport mode. No significant differences are apparent between each month for any transport mode. The sole observed pattern is that most bike trips tend to occur during summer months and in September. Concerning trips on weekdays, fewer public transportation trips appear to take place on weekends, whereas bike trips are less frequent on Sundays.

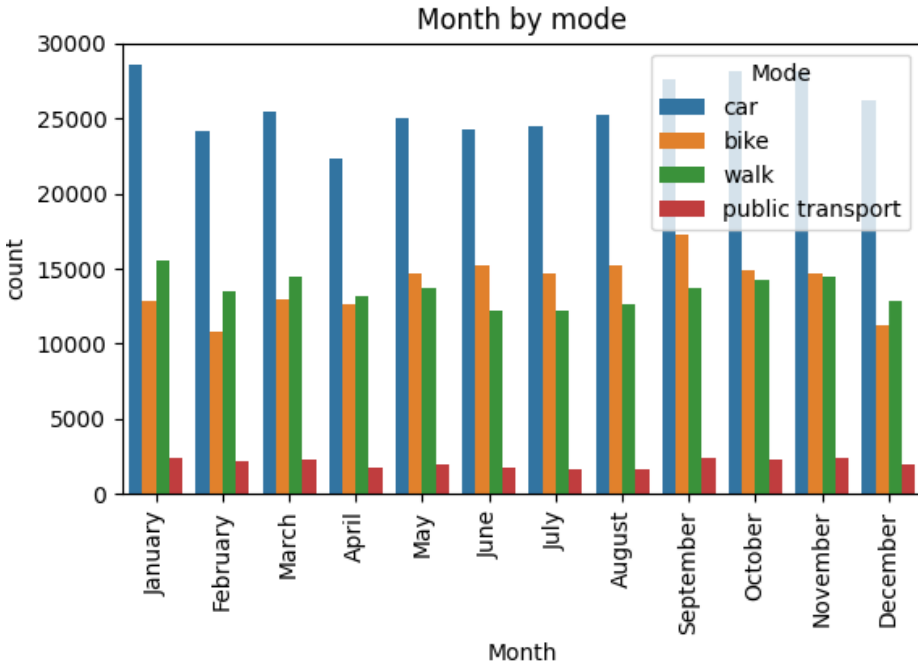


Figure 123. Month by mode

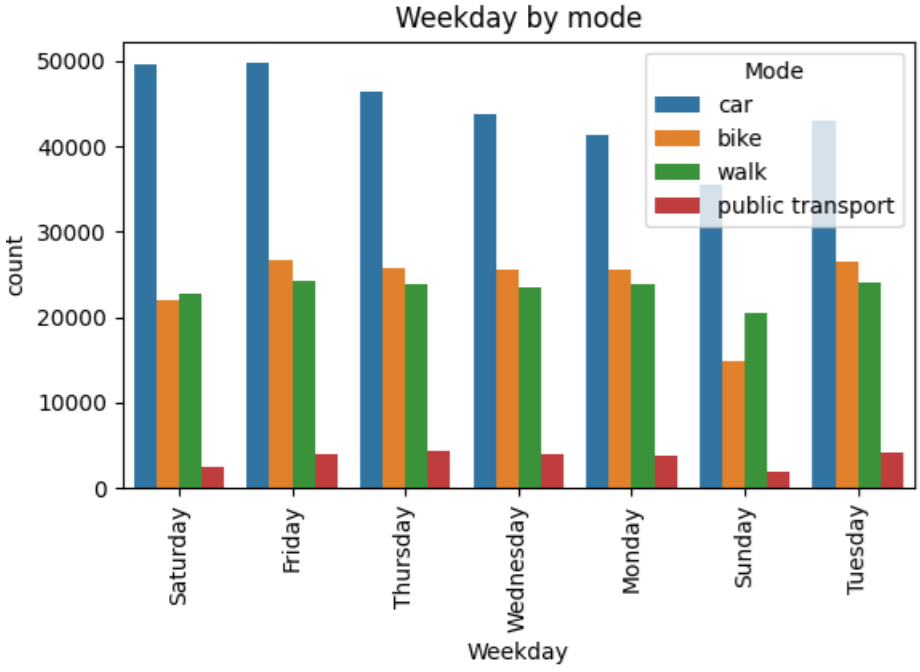


Figure 124. Weekday by mode

4.2.3. Data Preprocess

The initial step was to group low frequencies in some of the features. For education, “other education”, “primary education” and “no education” categories were grouped into “Other” category. The remaining categories along with their percentages are depicted below.

Education	
Higher education	48.513812
Secondary	32.915352
Lower vocational	13.775170
Other	4.795667

The same procedure was implemented for Motive feature. Service, education and Business-related trips were grouped into one category. The list of the new categories for the feature is illustrated below.

Motive	
Shopping/Grocery	21.314963
Go to/return from work	19.751561
Social and recreational	16.179105
Other motives	11.590445
Tours/hiking	11.369728
Visits	10.200755
Service/Education/Business	9.593443

Likewise, Hour representing the depart hour of the trip was grouped using the following procedure:

```
value_mapping = {0: "22.00-01.00",
                 1: "01.00-04.00",
                 2: "01.00-04.00",
                 3: "01.00-04.00",
                 4: "04.00-07.00",
                 5: "04.00-07.00",
                 6: "04.00-07.00",
                 7: "07.00-10.00",
                 8: "07.00-10.00",
                 9: "07.00-10.00",
                 10: "10.00-13.00",
                 11: "10.00-13.00",
                 12: "10.00-13.00",
                 13: "13.00-16.00",
                 14: "13.00-16.00",
                 15: "13.00-16.00",
                 16: "16.00-19.00",
                 17: "16.00-19.00",
                 18: "16.00-19.00",
                 19: "19.00-22.00",
                 20: "19.00-22.00",
                 21: "19.00-22.00",
                 22: "22.00-01.00",
                 23: "22.00-01.00"}
```

```
value_mapping = {"01.00-04.00":1,
                 "04.00-07.00":2,
                 "07.00-10.00":3,
                 "10.00-13.00":4,
                 "13.00-16.00":5,
                 "16.00-19.00":6,
                 "19.00-22.00":7,
                 "22.00-01.00":8}
df['Hour'] = df['Hour'].replace(value_mapping)
```

After grouping of lower categories, encoding into numerical form for all variables was implemented. Motive, was initially encoded using one hot encoding method, creating binary features (0, 1) for every possible category. Hence, 7 new features were created for the dataset as illustrated below.

Motive_Go to/return from work	Motive_Other motives	Motive_Service/Education/Business	Motive_Shopping/Grocery	Motive_Social and recreational	Motive_Tours/hiking	Motive_Visits
0	0	0	1	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	1
0	0	0	0	0	0	1
0	0	0	1	0	0	0
...	...	...	...	...	...	...
0	0	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	1	0	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0

Additionally, binary features were encoded using label encoder assigning 0 and 1 values. The process is depicted below.

```
le=LabelEncoder()
features=["Gender","Electric_bike", "Driver_license", "Round_trip", "Holiday", "Background"]
for col in features:
    df[col]=le.fit_transform(df[col])
```

Weekdays were encoded from 1-7 using the following procedure.

Monday:1

Tuesday:2

Wednesday:3

Thursday:4

Friday:5

Saturday:6

Sunday:7

Months were also encoded into 1-12 using the following procedure.

January:1

February:2

March:3

May:4

April:5

June:6

July:7

August:8

September:9

October:10

November:11

December:12

The target variable was encoded with the following procedure:

Car: 0, Bike: 1, Walk: 2, Public Transport: 3

Finally, the encoding for education was:

Other: 1

Secondary: 2

Lower vocational: 3

Higher Education: 4

Now that all variables were encoded into numeric form, the correlation matrix was created to explore the relationship between the variables. The matrix is depicted below.

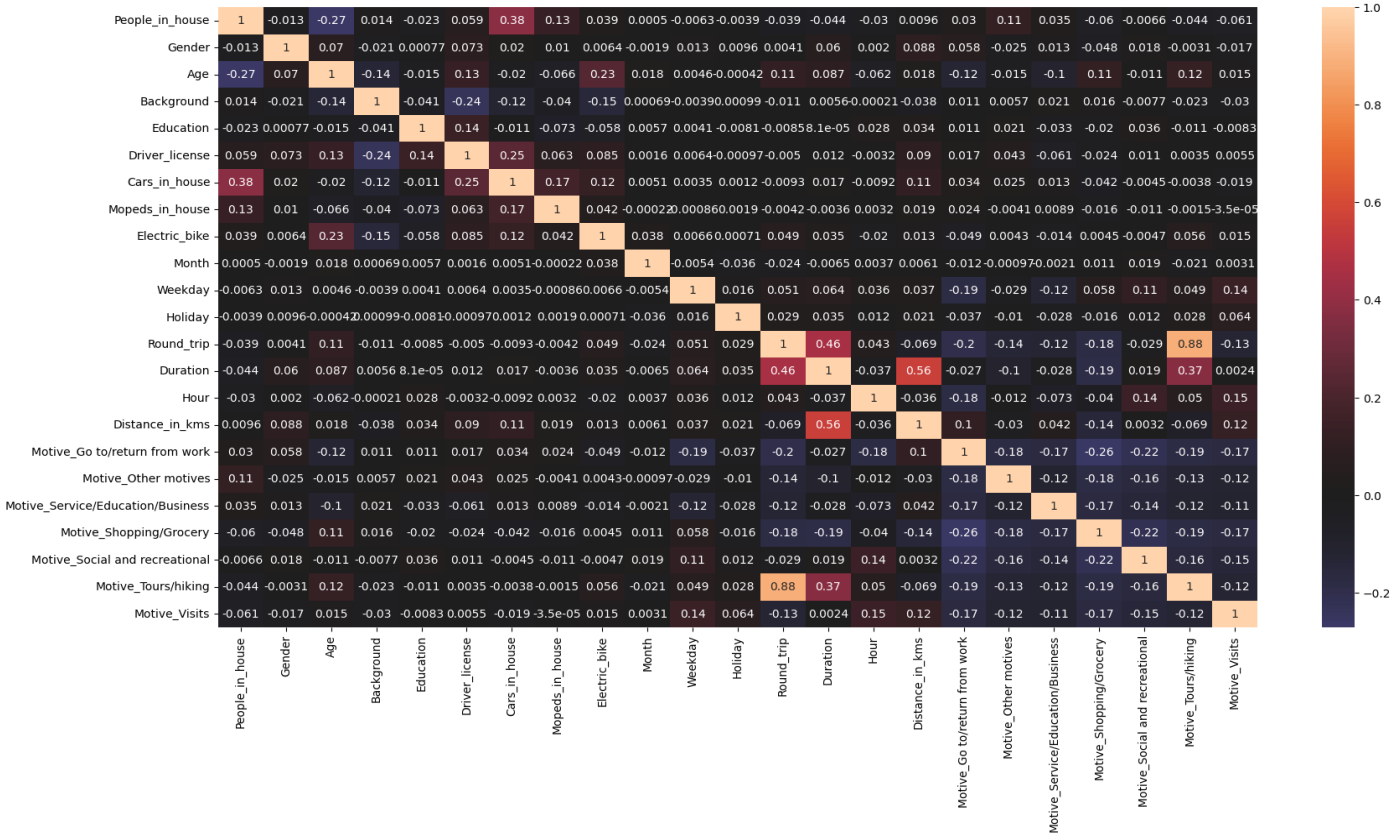


Figure 125. Correlation Matrix

From the matrix, it is identified that the highest correlation exists between Round_trip and Motive_Tours/hiking features with 0.88. The second highest correlation exists between Distance and Duration with 0.56, followed by Duration and Round_trip with correlation of 0.46. While the correlation matrix is valuable for identifying correlated features, the Cramer's V correlation matrix was also computed to explore the association between categorical variables. The values of the correlation range from 0 to 1 with values close to 1 indicating high association among the features (Zach, 2021). The new correlation matrix is illustrated below.

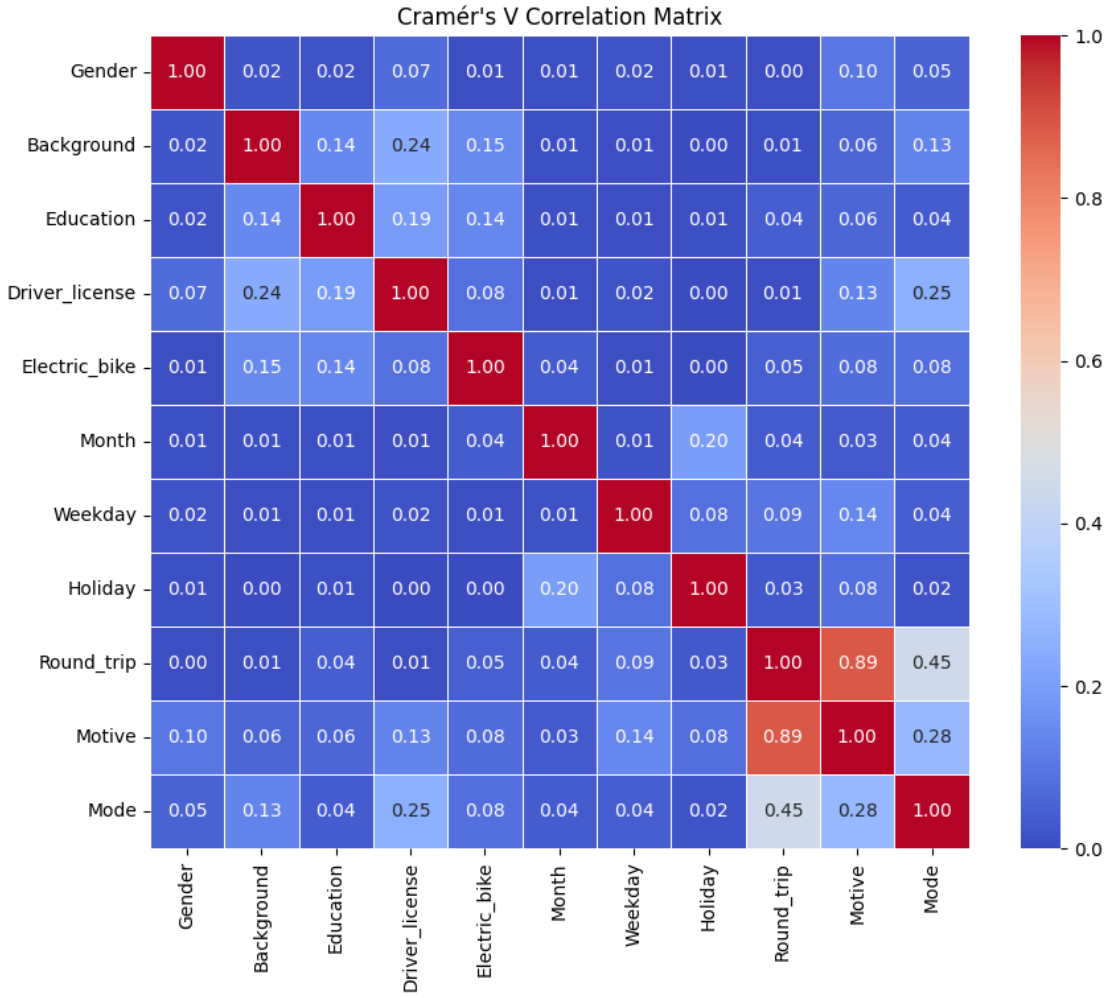


Figure 126. Cramer's V correlation matrix

From the plot above, it is identified that motive has a strong association with round trip highlighting a value of 0.89. The second strongest association exists between transport mode and round trip with a value of 0.45. All other associations between the categorical variables are low with values below 0.3.

To access whether there is a multicollinearity issue, the VIF factor was computed for all the predictor variables. The results of the first try are depicted below.

	Variable	VIF
0	const	0.000000
1	People_in_house	1.305560
2	Gender	1.022053
3	Age	1.250208
4	Background	1.093347
5	Education	1.036960
6	Driver_license	1.172424
7	Cars_in_house	1.301553
8	Mopeds_in_house	1.048903
9	Electric_bike	1.098639
10	Month	1.004095
11	Weekday	1.084131
12	Holiday	1.007861
13	Round_trip	5.830478
14	Duration	2.499606
15	Hour	1.098933
16	Distance_in_kms	2.158780
17	Motive_Go to/return from work	inf
18	Motive_Other motives	inf
19	Motive_Service/Education/Business	inf
20	Motive_Shopping/Grocery	inf
21	Motive_Social and recreational	inf
22	Motive_Tours/hiking	inf
23	Motive_Visits	inf

It is identified that Round_trip demonstrates a high VIF value of 5.83. Additionally, the motive features, which were one hot encoded, demonstrate a value of “inf” indicating multicollinearity. Hence, round_trip and Morive_Tours/hiking, demonstrating a correlation of 0.88 and an association of 0.89, were removed and VIF was computed again. The results are depicted below.

	Variable	VIF
0	const	77.984137
1	People_in_house	1.305347
2	Gender	1.021973
3	Age	1.249790
4	Background	1.093342
5	Education	1.036946
6	Driver_license	1.172353
7	Cars_in_house	1.301313
8	Mopeds_in_house	1.048903
9	Electric_bike	1.098632
10	Month	1.003901
11	Weekday	1.083643
12	Holiday	1.007859
13	Duration	2.235803
14	Hour	1.098616
15	Distance_in_kms	2.022188
16	Motive_Go to/return from work	3.384759
17	Motive_Other motives	2.481889
18	Motive_Service/Education/Business	2.241685
19	Motive_Shopping/Grocery	3.314130
20	Motive_Social and recreational	2.645792
21	Motive_Visits	2.243883

Figure 127. VIF - Case 2

Since all the VIF values are now below 5, those will be the final features for the model construction. The next step was to reduce some of the variability on the dataset. Recall, that in exploratory data

analysis chapter, distance and duration demonstrated an increased variability as also identified in the respective boxplots. To deal with such variability, instances were removed for **Duration >= 80 minutes** and **Distance >=75 kilometres**. The change in the histograms and boxplots of the features is illustrated below.

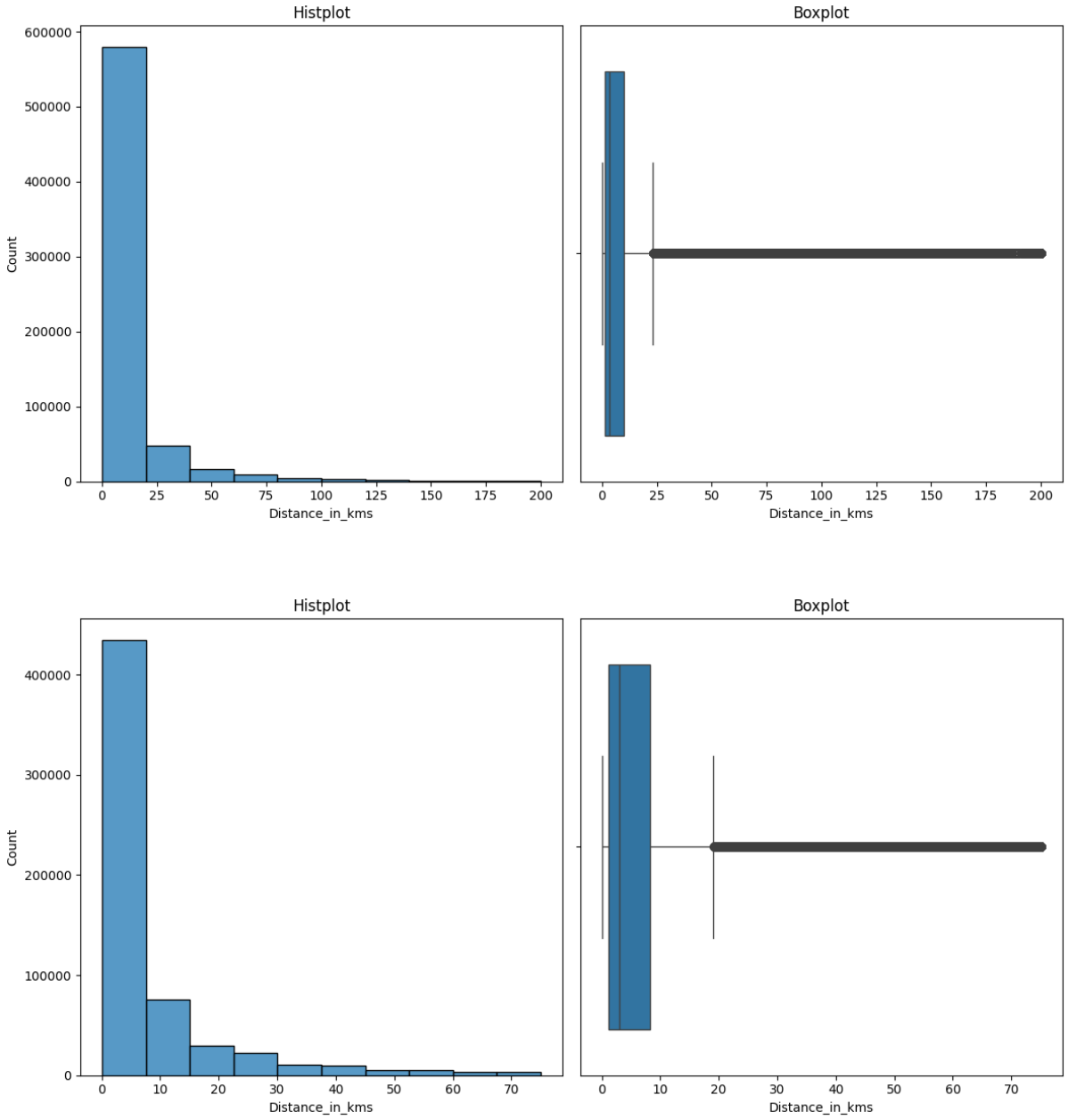


Figure 128. New Distance plots

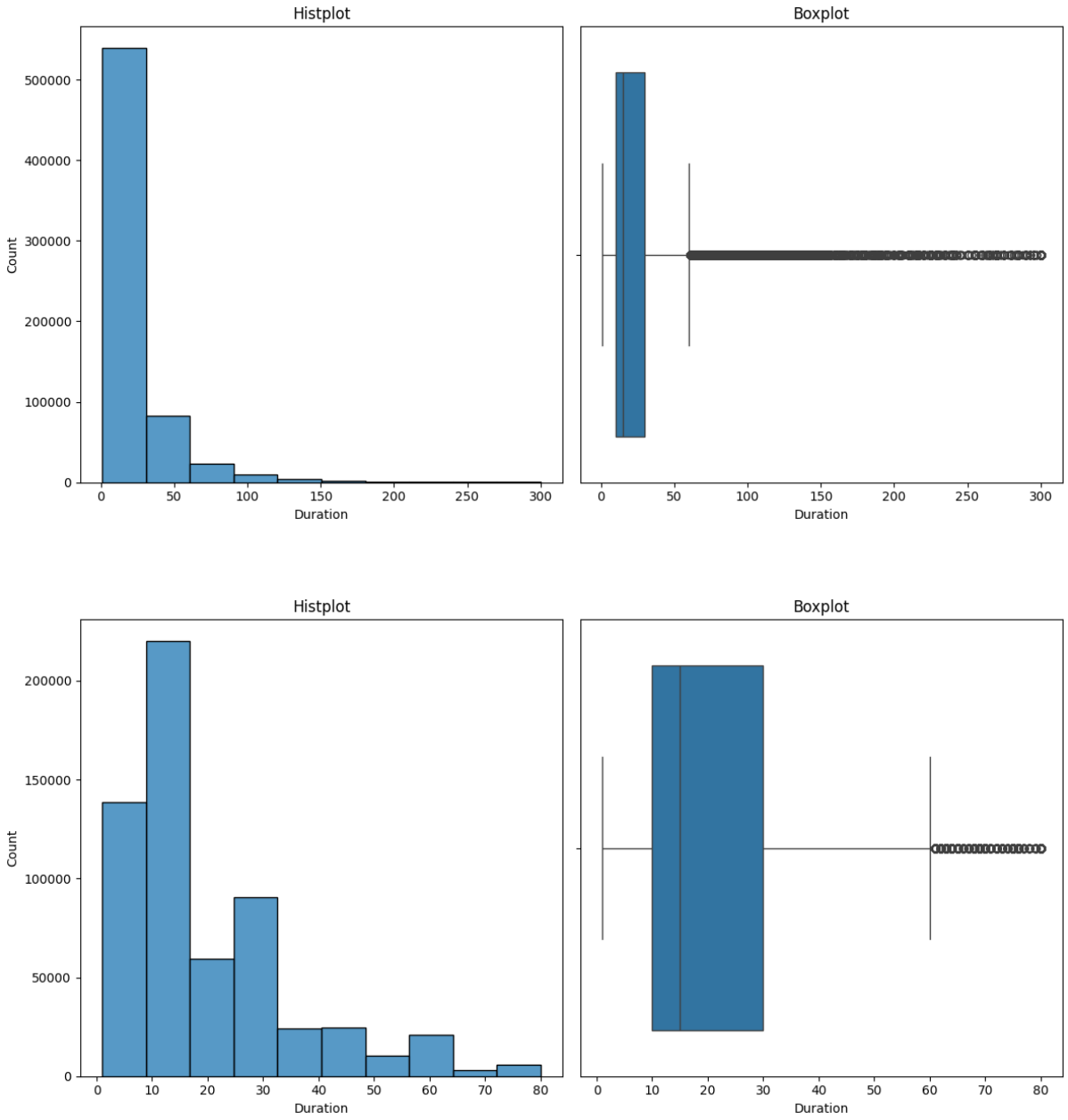


Figure 129. New duration plots

The new dataset shape consists of 597567 samples and 21 features. The last step was to shuffle the dataset and split it into train and test sets. The split ratio was 90:10, indicating that 90% of the data will train the model, and 10% of the data will be used for testing. The stratify option was used for splitting, so the imbalanced distribution to be preserved for both train and test sets. Below the distribution for each class is depicted for both training and testing.

```
Class counts in y_train:
Mode
0    249492
1    134778
2    131900
3     21640
Name: count, dtype: int64

Class counts in y_test:
Mode
0    27722
1    14975
2    14656
3     2404
Name: count, dtype: int64
```

4.2.4. Dummy Classifier

Prior to model application, a dummy was created using the DummyClassifier() function of sklearn. The purpose of the dummy classifier is to generate predictions without capturing any relationship on the data, just by random guessing. The classifier is particularly useful on heavily imbalanced datasets, serving as a reference point or a benchmark to compare with other classifiers. Three strategies were selected to generate predictions: Most frequent, Uniform and Stratified.

Most frequent: The dummy will generate predictions based on the most frequent class label (car in this instance)

Uniform: The dummy will generate predictions completely random

Stratified: The dummy will generate predictions based on the class distribution.

The dummies were trained on the training data and predictions were made using the test set. The results for the three dummies are depicted below.

Table 41. Dummy classifier metrics

Dummy-Most Frequent	Precision	Recall	F1	Accuracy
Car	0.46	1.00	0.63	0.46
Bike	0.00	0.00	0.00	
Walk	0.00	0.00	0.00	
Public transport	0.00	0.00	0.00	
Macro Average	0.12	0.25	0.16	

Dummy-Uniform	Precision	Recall	F1	Accuracy
Car	0.46	0.25	0.32	0.25
Bike	0.25	0.25	0.25	
Walk	0.24	0.25	0.24	
Public transport	0.04	0.26	0.07	
Macro Average	0.25	0.25	0.25	

Dummy-Stratified	Precision	Recall	F1	Accuracy
Car	0.46	0.46	0.46	0.34
Bike	0.25	0.25	0.25	
Walk	0.24	0.24	0.24	
Public transport	0.04	0.04	0.04	
Macro Average	0.25	0.25	0.25	

The confusion matrices were also constructed for the three dummies. The normalized versions were constructing (row normalization) demonstrating the recall proportion for each class. The results are illustrated below.

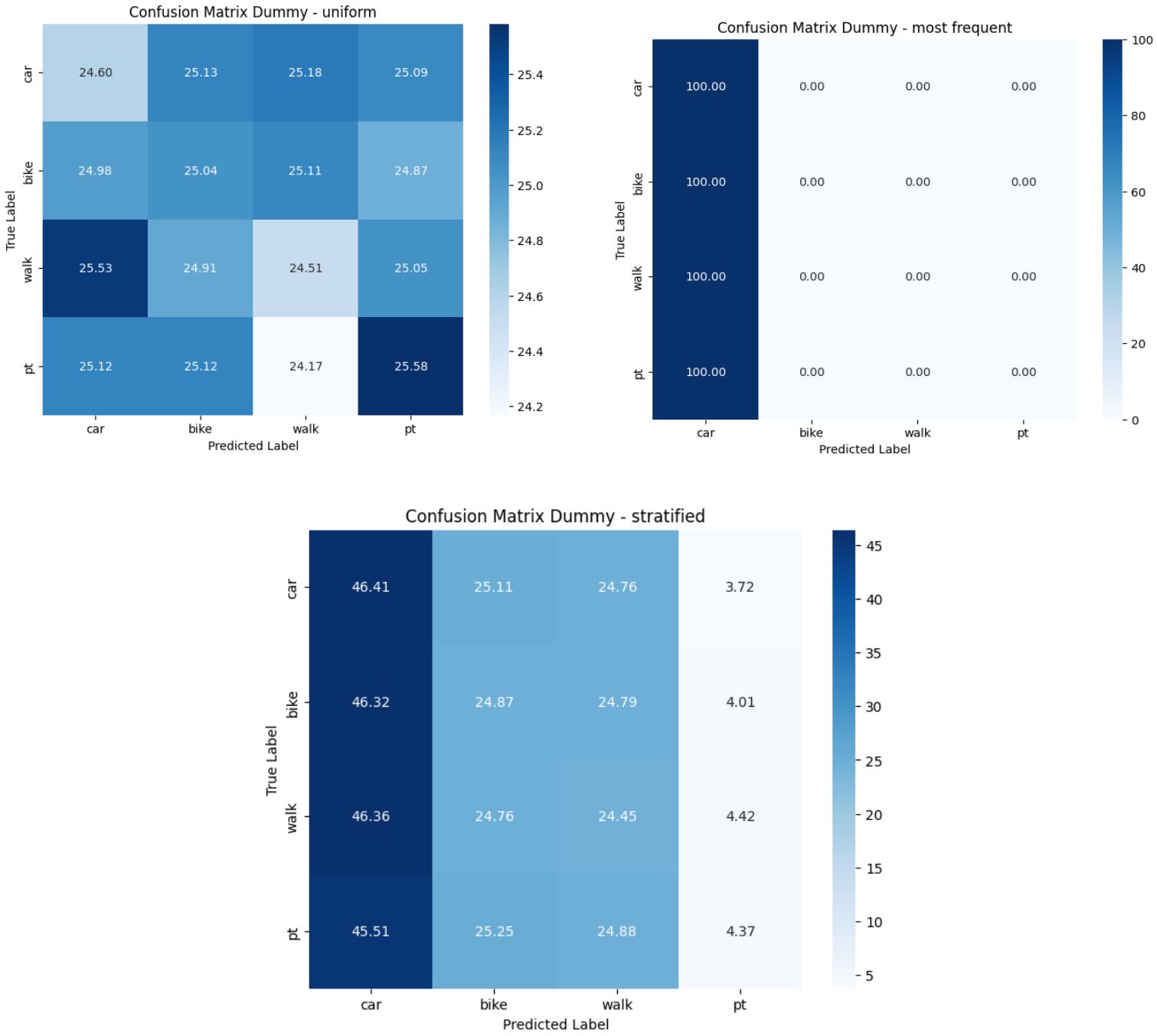
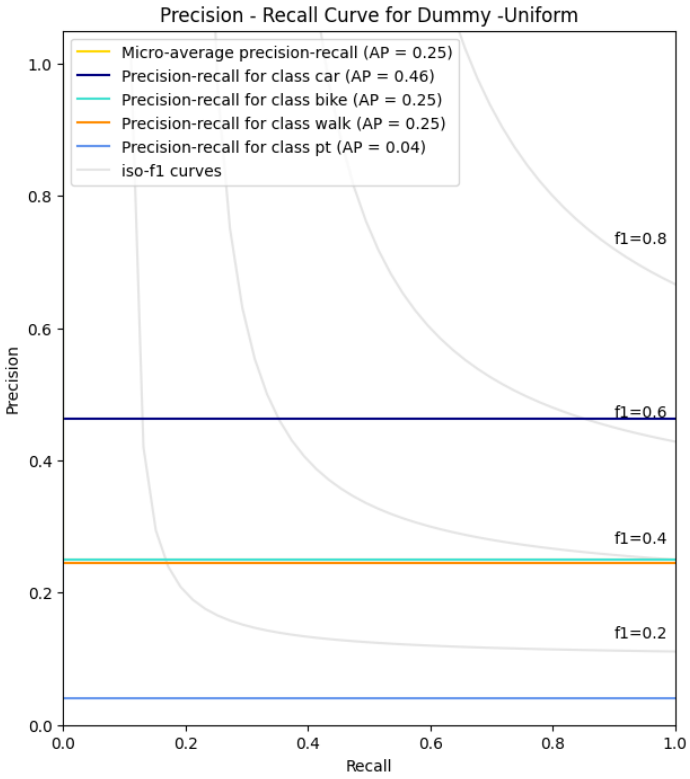
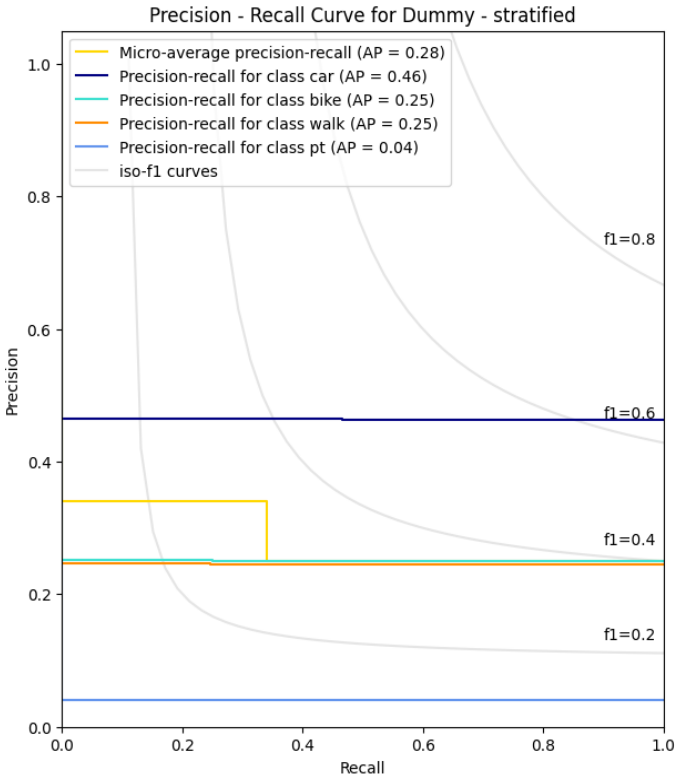


Figure 130. Dummy classifier - Confusion Matrices

Finally, the precision – recall curves were constructed to see the performance for each class on the dummies and primarily on the minority (public transport).





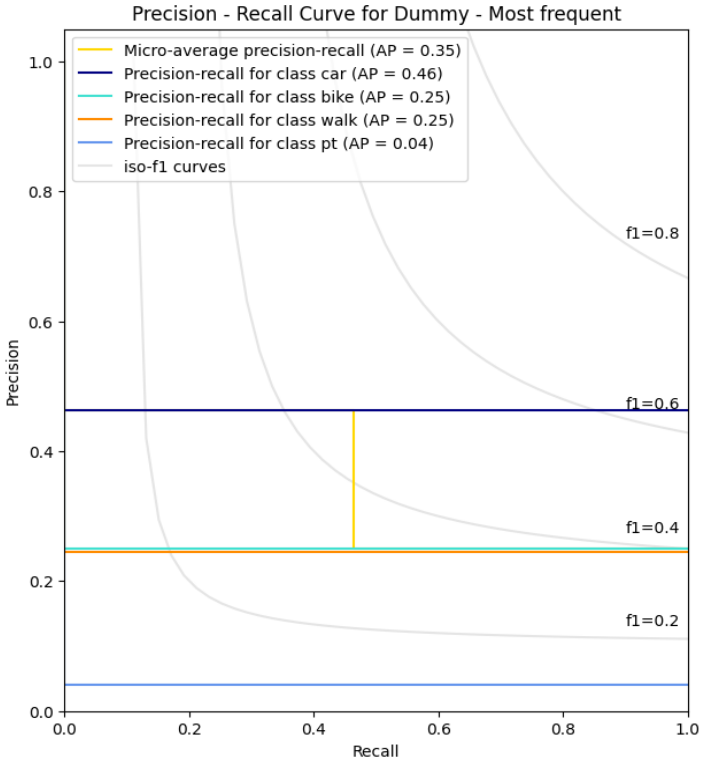


Figure 131. Dummy classifier Precision - Recall Curves

The results indicate that the dummies favour car label, while the performance for public transport is the worst. Bike and walk labels are also demonstrating low performance compared to car. Those results will serve as a benchmark to compare the following models.

4.2.5. Logistic Regression

The first model that was applied was that of Logistic Regression. The LogisticRegression(multi_class='ovr') function of sklearn was used to create the model. The model was fit on the training without any resampling on the training set. The model was then evaluated on the test set. The results are demonstrated on the table below.

Table 42. Logistic Regression Default

Logistic - Default	Precision	Recall	F1	Accuracy
Car	0.77	0.90	0.83	0.74
Bike	0.61	0.47	0.53	
Walk	0.77	0.82	0.79	
Public transport	0.50	0.05	0.08	
Macro Average	0.66	0.56	0.56	

The overall accuracy of the model is 0.74. The best performances are for Car and Walk labels demonstrating recalls of 0.90 and 0.82 respectively and a precision of 0.77 each. Performance for bike is mediocre with precision 0.61 and a recall of 0.47. The performance of public transport, which is the minority class, is poor with precision of 0.50 and a very low recall of 0.05 indicating the inability of the model to retrieve relevant instances for the minority class. This outcome is a consequence of the training model being biased due to the severity of the imbalance. First, an undersampling strategy was implemented on the training set to combat class imbalance. A python pipeline was implemented from imblearn library to automate the undersampling strategy and combine it with parameter tuning-cross validation and normalization on the data. The pipeline configuration is depicted below.

```
pipeline = Pipeline(steps = [
   ('scaler', MinMaxScaler()),
   ('sampler', RandomUnderSampler(sampling_strategy='auto', random_state=42)),
   ('classifier', LogisticRegression(multi_class='ovr'))
])
param_grid = {'classifier__solver' : ['liblinear', 'lbfgs'],
          'classifier__max_iter': [400]}
grid_search_log = GridSearchCV(pipeline, param_grid, cv=skf, scoring='f1_macro')
grid_search_log.fit(X_train, y_train)
```

First the pipeline will normalize the training set into 0, 1 scale, and undersample the training set. The sampling strategy set to "auto" indicates that samples will be removed from all classes until they equalize the minority class (public transport). Afterwards, the logistic regression model will be fitted on the train set. This process occurs for every iteration of cross validation. More specifically, since 10

folds are used, the training set is split into 10 equal folds. The undersampling of the data will occur only for the 9 folds that will train the model for every possible parameter combination, while the 10th validation fold remains unbalanced. The StratifiedKfold function was used to create the 10 folds, ensuring that each fold has approximately the same class distribution and samples from the minority class are included in each fold. The f1-macro was used as the scoring metric for the validation process since both precision and recall are desirable. The "macro" approach is the average of the f1 scores from the individual classes, giving equal importance to the classes regardless of the imbalance. In imbalanced situations, it is often preferred as it provides a more practical performance for the classifier, ensuring that the minority class contributes equally to the majority classes.

The best validation results of the undersampled model are for **solver: 'lbfgs'** and a **validation f1 macro of 0.57**. The model was then retrieved and evaluated on the test set. Note, that the pipeline is functioning in such way that only normalization occurs on the test set and not resampling, thus the test set remains unbalanced. The results are depicted below.

Table 43. Logistic Regression - Undersampling

Logistic - Undersampling	Precision	Recall	F1	Accuracy
Car	0.88	0.70	0.78	0.65
Bike	0.51	0.48	0.50	
Walk	0.75	0.74	0.74	
Public transport	0.17	0.62	0.27	
Macro Average	0.58	0.64	0.57	

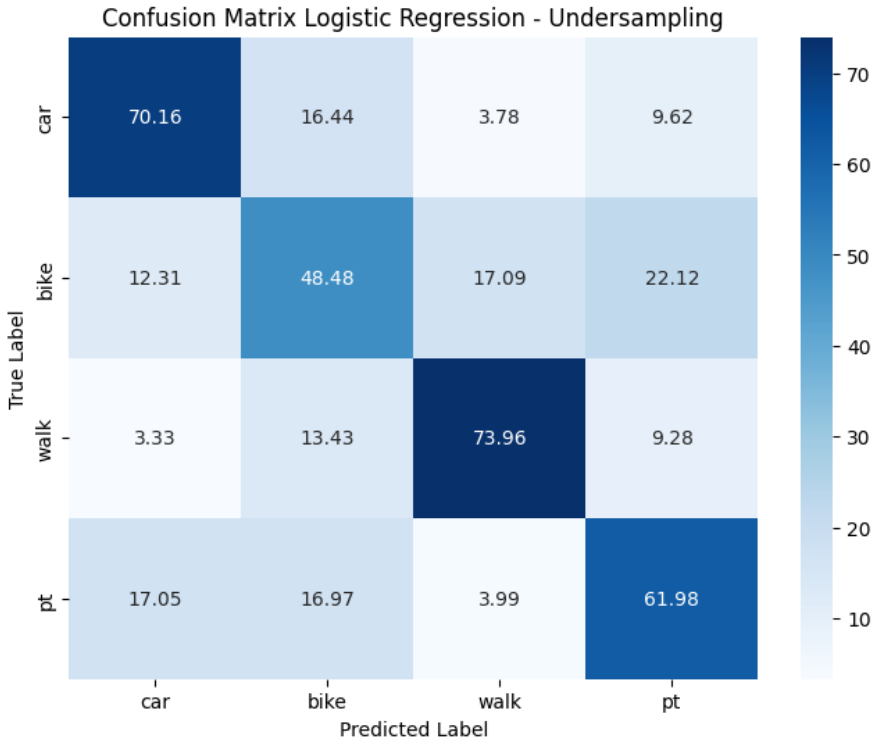


Figure 132. Logistic Regression - Confusion Matrix - Undersampling

The overall accuracy of the model is 0.65. Recall for the minority class has significantly increased compared to the default model, as it increased from 0.05 to 0.62. Recall has slightly increased for bike as well from 0.47 to 0.48 though performance is still below average, as precision remains low at 0.51. In contrast, recalls decreased for car and walk at 0.70 and 0.74 respectively. Although the overall accuracy of the model decreased from 0.74 to 0.65, the undersampled model demonstrates a more balanced performance since it captures significantly higher relevant instances for the minority class. The overall performance though, is still low for the model.

The second experimentation was with an Oversampling strategy. Random samples are duplicated from the minority classes until they match those of the majority. To ensure that oversampling applies only on the training set, while cross validation and tuning applies at the same time, the pipeline was used again. The configuration of the pipeline is illustrated below.

```
pipeline = Pipeline(steps = [
   ('scaler', MinMaxScaler()),
   ('sampler', RandomOverSampler(sampling_strategy='auto', random_state=42)),
   ('classifier', LogisticRegression(multi_class='ovr'))
])
param_grid = {'classifier__solver' : ['liblinear', 'lbfgs'],
          'classifier__max_iter': [400]}
grid_search_log = GridSearchCV(pipeline, param_grid, cv=skf, scoring='f1_macro')
grid_search_log.fit(X_train, y_train)
```

The same process is applied as the previous model with the under sampling strategy. The validation results are again for **solver: 'lbfgs' and a validation f1 macro score of 0.59**. The best estimator was then retrieved and evaluated on the test set. The results are depicted below.

Table 44. Logistic Regression Oversampling

Logistic - Oversampling	Precision	Recall	F1	Accuracy
Car	0.88	0.71	0.78	0.67
Bike	0.54	0.47	0.50	
Walk	0.76	0.83	0.79	
Public transport	0.18	0.62	0.28	
Macro Average	0.59	0.66	0.59	

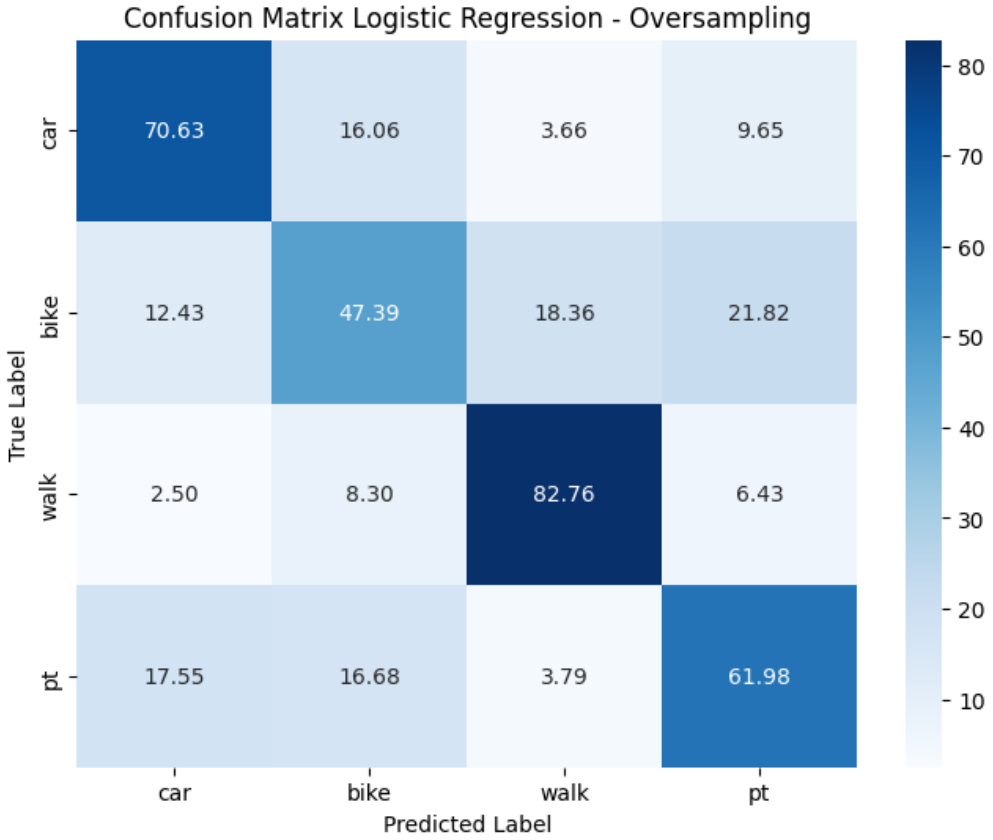


Figure 133. Logistic Regression - Confusion Matrix - Oversampling

Comparing the performance with the undersampled model, there are minor changes in the performance. The overall accuracy of the model increased from 0.65 to 0.67. There is a significant change in recall for walk class, as it increased from 0.74 to 0.83, while the f1 score increased from 0.74 to 0.79. For the public transport the metrics are identical with a minor increase in f1 score from 0.27 to 0.28, though the performance remains relatively low. Metrics for the other two classes are also quite similar regardless of resampling strategy. As a last evaluation for the model, the precision - recall (PR) curves were constructed for the undersampled and oversampled models. The PR is most suitable for cases with a severe class imbalance. To assess the model's performance and the classes, the area under the curve (average precision) is calculated, ranging from 0 to 1. Higher values indicate better class performance. The micro-AP is also computed which is the weighted average of the area for all the classes (Kashifi et al., 2022). The curves for both models are depicted below.

There is a slight increase in performance using oversampling for most of the classes. The AP for car remained the same while the highest change occurs for walk as AP increased from 0.84 to 0.87. For bike AP also increased from 0.48 to 0.50, while for the minority class, public transport, there is a minor increase from 0.27 to 0.28. The micro-AP of the model overall increased from 0.72 to 0.74. The performance is satisfactory only for walk and car classes, while for bike is average.

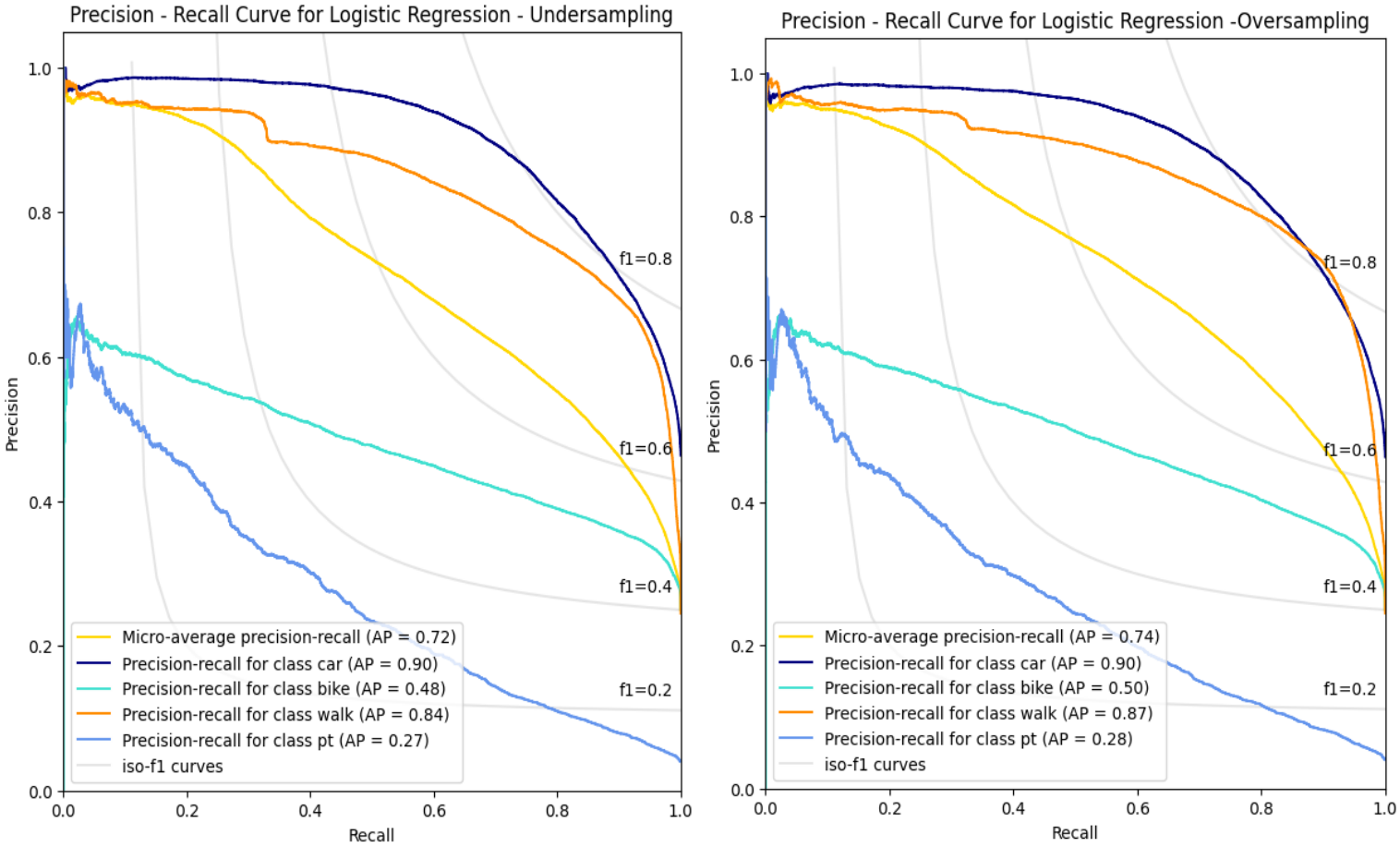


Figure 134. Precision Recall Curves - Logistic Regression

The performance is low for the minority class as the area under the curve remains low. Both models were compared to the corresponding undersampled and oversampled logistic regression models proposed by Kashifi et al, 2022.

Table 45. Model comparison - Logistic Regression - Undersampling

Logistic Regression - Undersampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.87	0.42	0.43	0.26
Recall	0.56	0.45	0.70	0.66
F1 - score	0.68	0.43	0.54	0.38
AP	0.88	0.45	0.55	0.32
Micro-AP	0.60			
F1 - Macro	0.5075			
Accuracy	0.566			

Logistic Regression - Undersampling **Thesis Model**				
	Car	Bike	Walk	Public Transport
Precision	0.88	0.51	0.75	0.17
Recall	0.80	0.48	0.74	0.62
F1 - score	0.78	0.50	0.74	0.27
AP	0.90	0.48	0.84	0.27
Micro-AP	0.72			
F1 - Macro	0.57			
Accuracy	0.65			

Table 46. Model comparison - Logistic Regression - Oversampling

Logistic Regression - Oversampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.89	0.43	0.44	0.24
Recall	0.58	0.46	0.73	0.69
F1 - score	0.70	0.44	0.55	0.35
AP	0.89	0.46	0.60	0.35
Micro-AP	0.62			
F1 - Macro	0.51			
Accuracy	0.58			

Logistic Regression - Oversampling **Thesis Model**				
	Car	Bike	Walk	Public Transport
Precision	0.88	0.54	0.76	0.18
Recall	0.71	0.47	0.83	0.62
F1 - score	0.78	0.50	0.79	0.28
AP	0.90	0.50	0.87	0.28
Micro-AP	0.74			
F1 - Macro	0.59			
Accuracy	0.67			

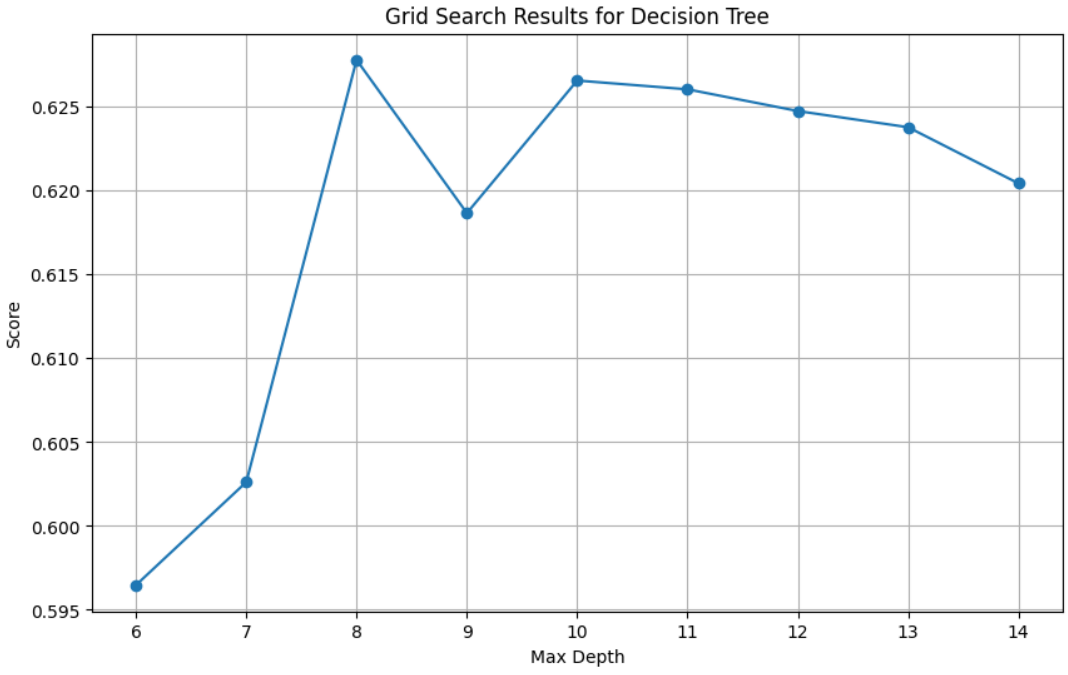
Compared to the models of Kashifi (2022), there is a lower performance for class “public transport” regardless of resampling strategy. In contrast there is a huge performance gap for class “walk” with all the metrics being higher. This also occurs for the metric in classes “car” and “bike”.

4.2.6. Decision Tree

The next model that was applied was that of the decision tree. Similarly to the previous model, both undersampling and oversampling techniques were used. The pipeline and GridSearchCV() functions were used again to ensure the resampling strategy occurs within the cross validation process and hyperparameter tuning. The parameter that was tuned for both resampling methods was that of the max depth, which controls the overall tree size. First the undersampling technique was used. The configuration of the model is depicted below. Stratified 10 k fold was used for each fold to have the same distribution, while f1 macro was used as the scoring option.

```
pipeline = Pipeline(steps = [
    ('scaler', MinMaxScaler()),
    ('sampler', RandomUnderSampler(sampling_strategy='auto', random_state=42)),
    ('classifier', DecisionTreeClassifier(random_state=42))
])
param_grid = {
    'classifier__max_depth': list(range(6, 15)),
}
grid_search = GridSearchCV(pipeline, param_grid, cv=skf, scoring="f1_macro", n_jobs=-1)
```

The cross-validation results for the undersampled tree are shown on the plot below.



The optimal **max_depth was found to be 8**, with a validation **F1- macro of 0.63**. The optimal tree was then evaluated on the test set. The results are illustrated on the classification report and the confusion matrix below.

Table 47. Decision Tree Undersampling

Tree - Undersampling	Precision	Recall	F1	Accuracy
Car	0.92	0.68	0.78	0.71
Bike	0.54	0.63	0.58	
Walk	0.78	0.86	0.82	
Public transport	0.25	0.55	0.34	
Macro Average	0.62	0.68	0.63	

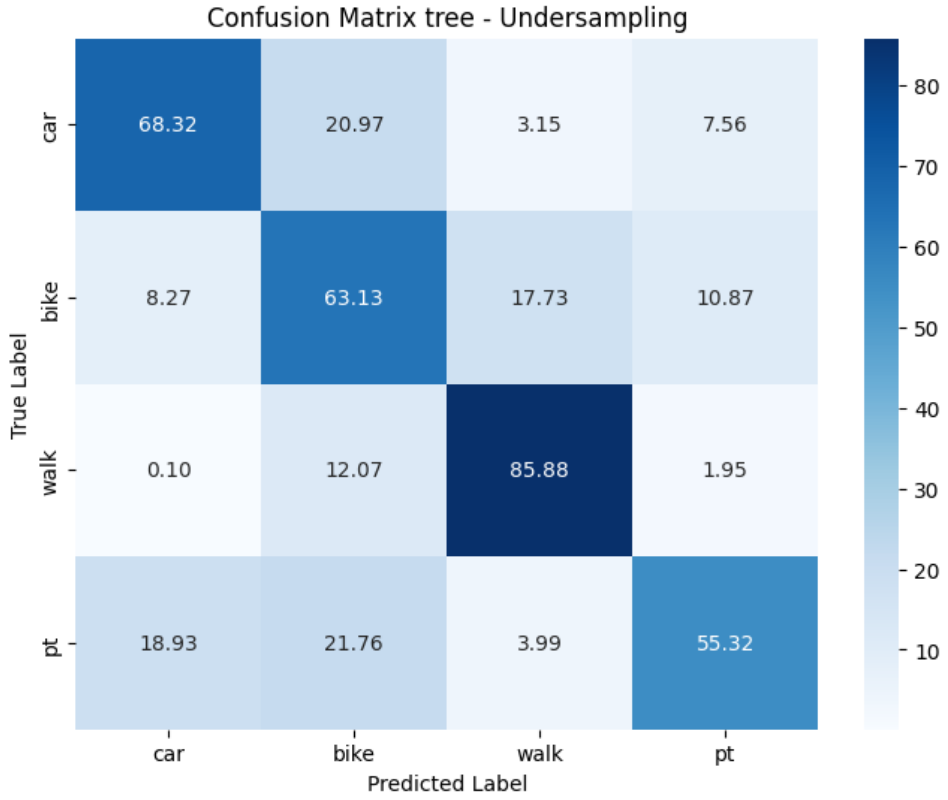


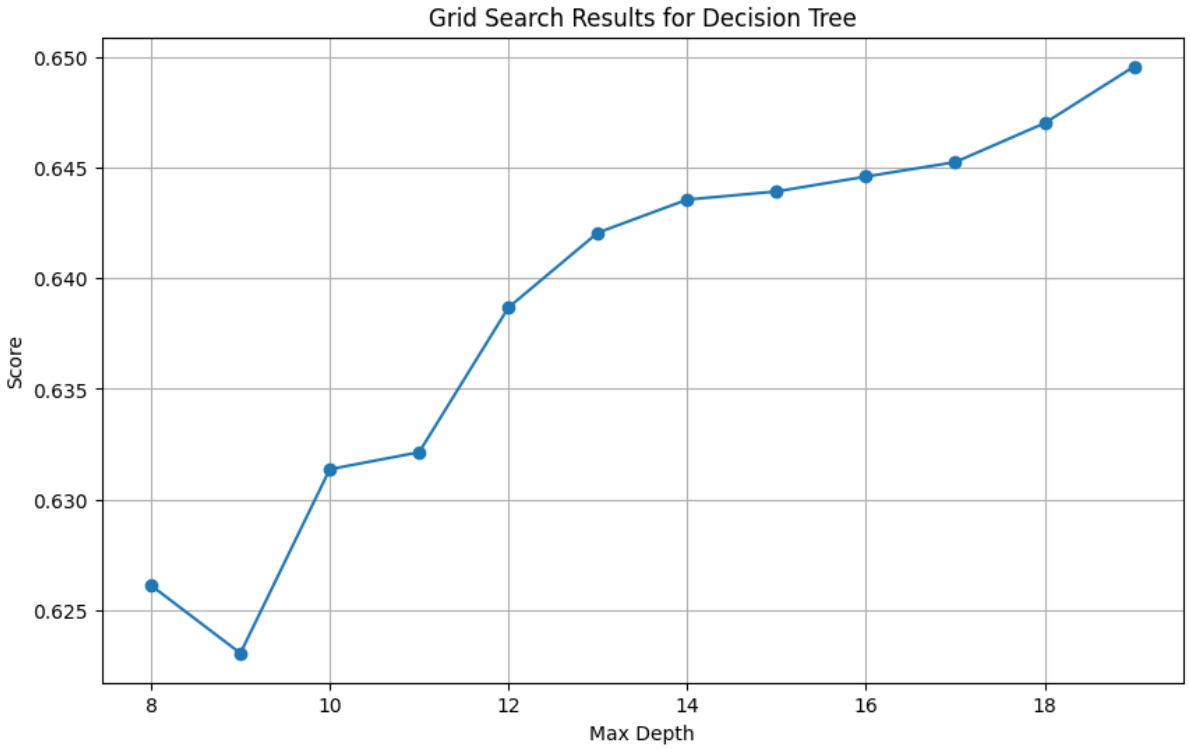
Figure 135. Decision Tree - Confusion Matrix - Undersampling

The best model performance is for class "walk" with a precision of 0.78, recall of 0.86 and an f1 score of 0.82. For the minority class, public transport, the model has a recall score of 0.55, capturing 55.32% of the true labels for the class. However, precision is low with 0.25 indicating, implying that of all the predictions the model made as public transport only 25% belonged in that class. F1 is also low for public

transport with 0.34. For car there is an average performance, with precision, recall and f1 score of 0.92, 0.68 and 0.78 respectively. Results are also average for “bike” with a f1 score of 0.58. The highest misclassification is between car and bike, since the model predicted 20.97% of instances where the true label was car, as bike.

The second experimentation was using the oversampling strategy in the pipeline. Similarly, the max_depth of the tree was tuned in the grid search function. The configuration for the oversampled tree can be viewed below.

```
pipeline = Pipeline(steps = [
    ('scaler', MinMaxScaler()),
    ('sampler', RandomOverSampler(sampling_strategy='auto', random_state=42)),
    ('classifier', DecisionTreeClassifier(random_state=42))
])
param_grid = {
    'classifier__max_depth': list(range(8, 20)),
}
```



The tree with **max_depth=19** yielded the best results for the oversampling strategy with a validation **F1-macro of 0.65**. The optimal tree was then evaluated on the test set. The classification report and the confusion matrix are illustrated below.

Table 48. Decision Tree Oversampling

Tree - Oversampling	Precision	Recall	F1	Accuracy
Car	0.90	0.74	0.81	0.73
Bike	0.58	0.68	0.63	
Walk	0.82	0.81	0.81	
Public transport	0.27	0.54	0.37	
Macro Average	0.64	0.69	0.65	

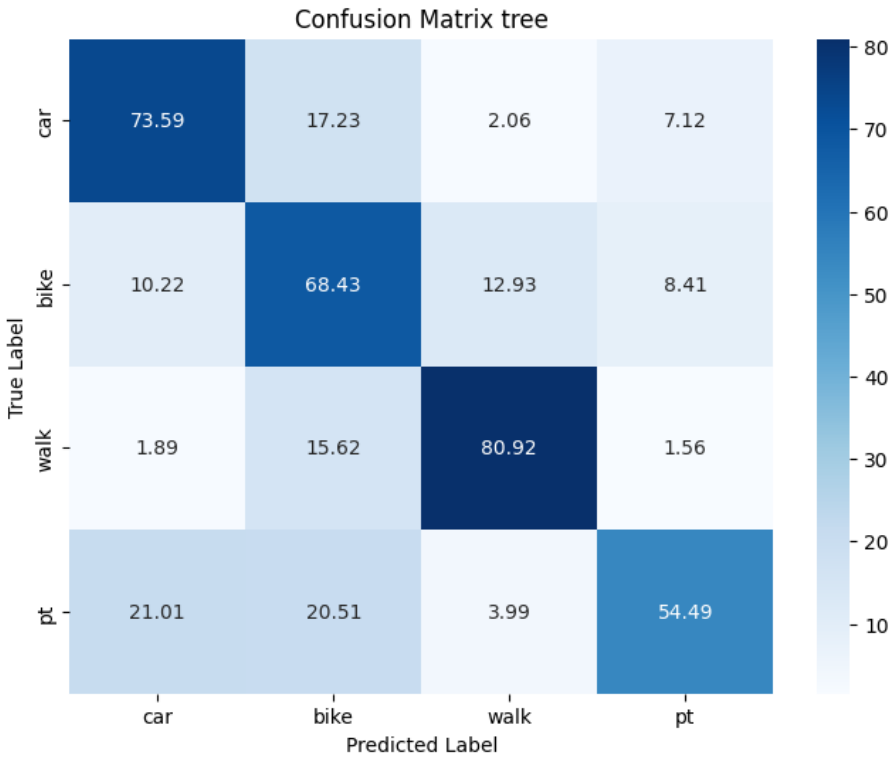


Figure 136. Decision Tree - Confusion Matrix - Oversampling

After implementing an oversampling strategy, the overall model accuracy improved from 0.71 to 0.73 compared to the undersampled model. The F1-macro also saw an increase from 0.63 to 0.65. All metrics exhibited improvements for each class, except for the recall in the "public transport" class, which experienced a minor decrease from 0.55 to 0.54. However, the F1 score remains at 0.27, indicating a low performance for the class. To further assess the performance of the resampled models, precision-recall curves were constructed, as depicted below.

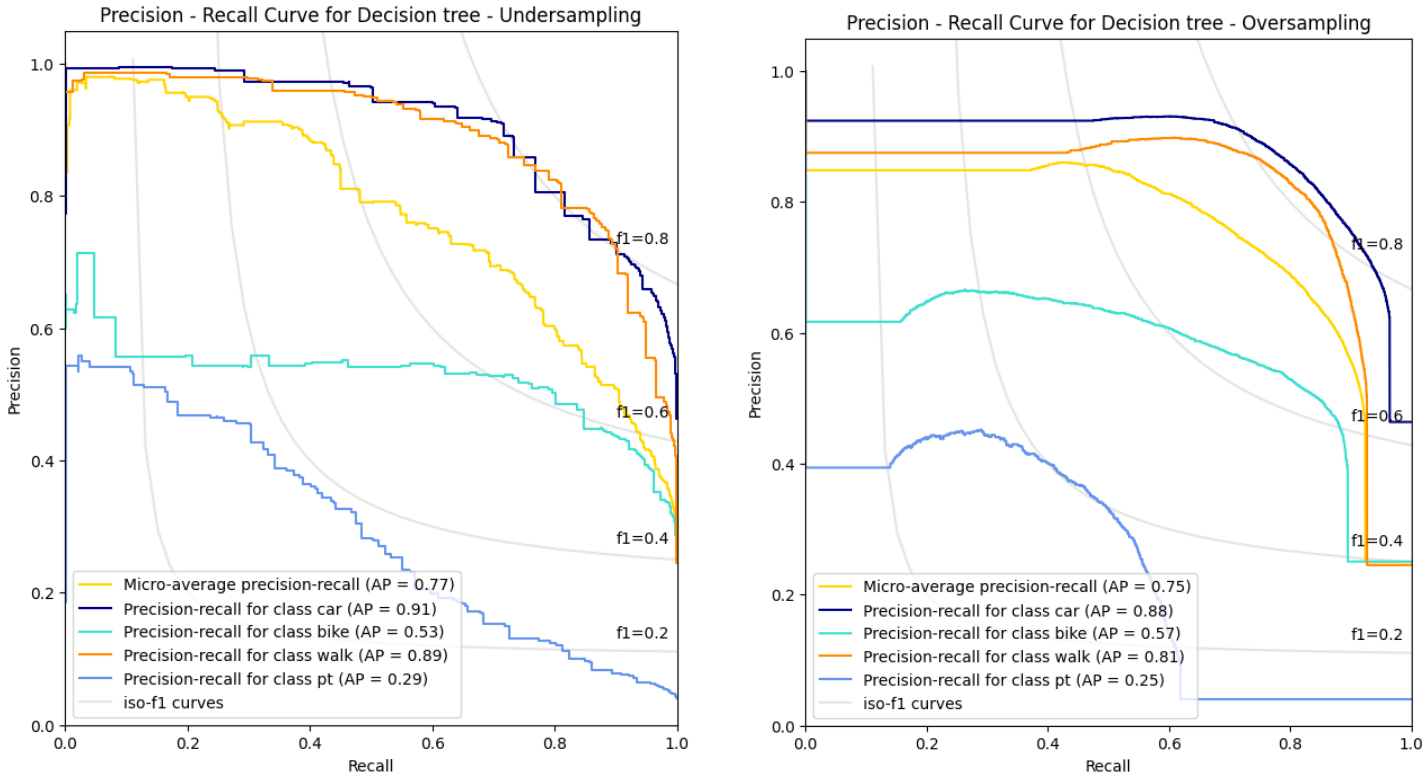


Figure 137. Precision Recall Curves - Decision Tree

As evident from the graphs, the undersampled tree demonstrates a better performance compared to oversampling. The micro AP is 0.77 compared to 0.75 with oversampling. Ap for car is 0.91 compared to 0.88 while for the minority class AP is 0.29 compared to 0.25 with oversampling. The only difference is for bike where AP with oversampling increased from 0.53 to 0.57. Overall, the undersampled Decision Tree performs better compared to the Oversampled Decision Tree. The feature importances were also extracted for the undersampled tree. The results are depicted below.

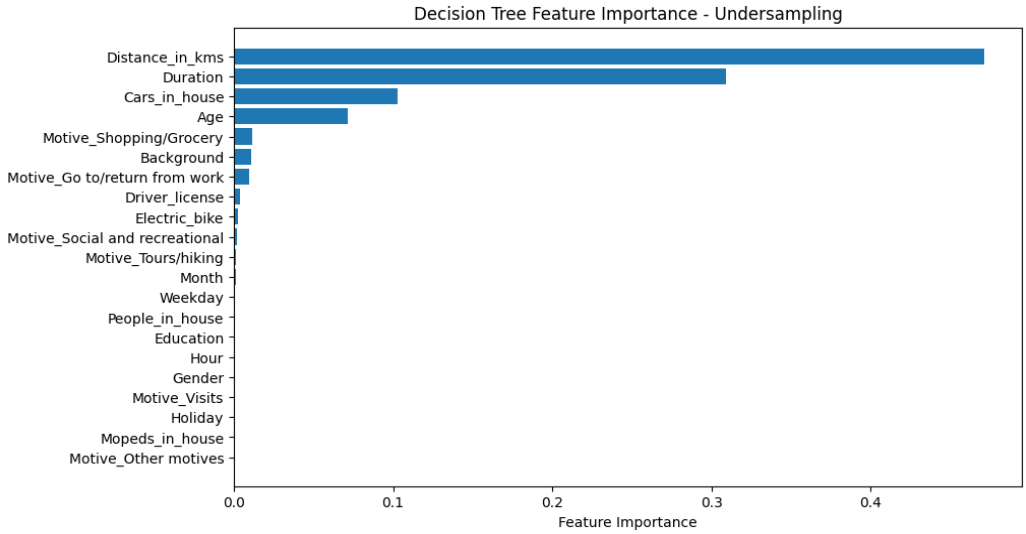


Figure 138. Tree Feature importance - Undersampling

The tree construction highlighted Distance, Duration, Age, and the number of cars in a household as the most crucial features. Like logistic regression models, the resampled Decision Trees were evaluated

against the corresponding models proposed by Kashifi et al. (2022). The tables displaying the performance metrics are presented below.

Table 49. Model comparison - Decision Tree - Undersampling

Decision Tree - Undersampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.83	0.47	0.47	0.22
Recall	0.52	0.53	0.63	0.76
F1 - score	0.66	0.49	0.54	0.34
AP	0.70	0.36	0.34	0.16
Micro-AP	0.43			
F1 - Macro	0.51			
Accuracy	0.56			

Decision Tree - Undersampling **Thesis - Model**				
	Car	Bike	Walk	Public Transport
Precision	0.92	0.54	0.78	0.25
Recall	0.68	0.63	0.86	0.55
F1 - score	0.78	0.58	0.82	0.34
AP	0.91	0.53	0.89	0.29
Micro-AP	0.77			
F1 - Macro	0.63			
Accuracy	0.71			

Table 50. Model comparison - Decision Tree - Oversampling

Decision Tree - Oversampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.87	0.73	0.74	0.50
Recall	0.87	0.75	0.71	0.49
F1 - score	0.87	0.74	0.72	0.50
AP	0.83	0.62	0.59	0.27
Micro-AP	0.70			
F1 - Macro	0.71			
Accuracy	0.80			

Decision Tree - Oversampling **Thesis - Model**				
	Car	Bike	Walk	Public Transport
Precision	0.90	0.58	0.82	0.27
Recall	0.74	0.68	0.81	0.54
F1 - score	0.81	0.63	0.81	0.37
AP	0.88	0.57	0.81	0.25
Micro-AP	0.75			
F1 - Macro	0.65			
Accuracy	0.73			

In the case of the undersampled model, the metrics for Accuracy, F1-macro, and Micro-Ap are consistently higher compared to the model proposed by Kashifi et al. (2022). Except for public transport, where metrics are slightly lower, the performance for the other classes is significantly

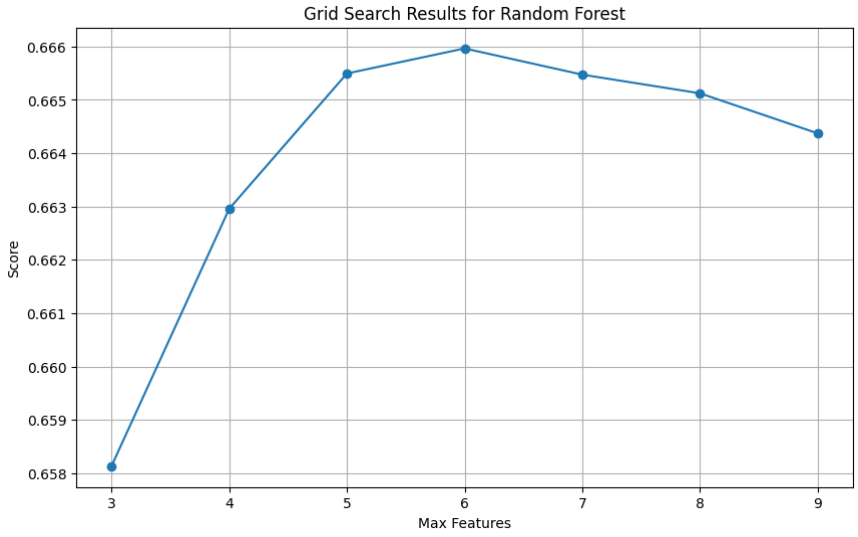
superior, particularly for "car" and "walk". The oversampled model exhibits lower performance in terms of Accuracy and F1-macro. On an individual class basis, the model proposed by Kashifi et al. (2022) outperforms in the "car," "bike," and "public transport" classes. However, the "walk" class shows significantly superior performance in the oversampled model, with all metrics of Precision, Recall, and F1 being higher for that class compared to the paper. Overall, the model of decision tree performs well for "car" and "walk" classes, while the performance is still low for "public transport" and average for "bike".

4.2.7. Random Forest

The next model employed was Random Forest, with the initial experiment involving an undersampling strategy. As with previous models, a pipeline was utilised in conjunction with GridSearch, ensuring that the model underwent resampling during the cross-validation process while parameter tuning took place. The first parameter subject to tuning was max_features, referring to the number of features considered in each split of the individual decision tree, with the value range set from 2 to 9 max features. The configuration of the undersampled model inside the pipeline is illustrated below.

```
pipeline = Pipeline(steps = [
    ('scaler', MinMaxScaler()),
    ('sampler', RandomUnderSampler(sampling_strategy='auto', random_state=42)),
    ('classifier', RandomForestClassifier(random_state=42, n_jobs=-1))
])
param_grid = {
    'classifier__max_features' : list(range(3, 10))
}
grid_search_rf = GridSearchCV(pipeline, param_grid, cv=skf, scoring='f1_macro')
```

The pipeline was then fitted on the training set. The results of the validation process are illustrated in the plot below.



The best validation results were for **max features =6**. Subsequently, a new grid search was conducted to explore combinations of both max_features and the number of estimators. The max features parameter was set at [6], while the number of trees ranged from [450, 500, 550]. The best result got for max features =6 and number of estimators of 550 with a validation f1 macro of 0.67. The best model was then retrieved and evaluated on the test set. The results are depicted below.

Table 51. Random Forest Undersampling

Random Forest Undersampling	Precision	Recall	F1	Accuracy
Car	0.93	0.73	0.82	0.74
Bike	0.61	0.64	0.63	
Walk	0.82	0.86	0;.84	
Public transport	0.28	0.78	0.41	
Macro Average	0.66	0.75	0.67	

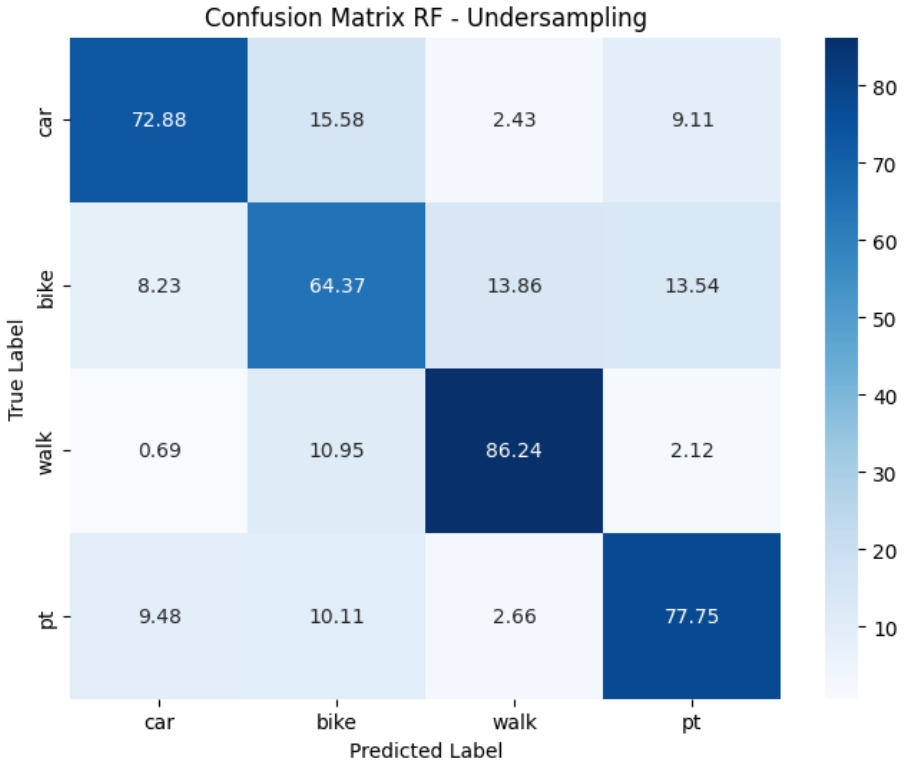


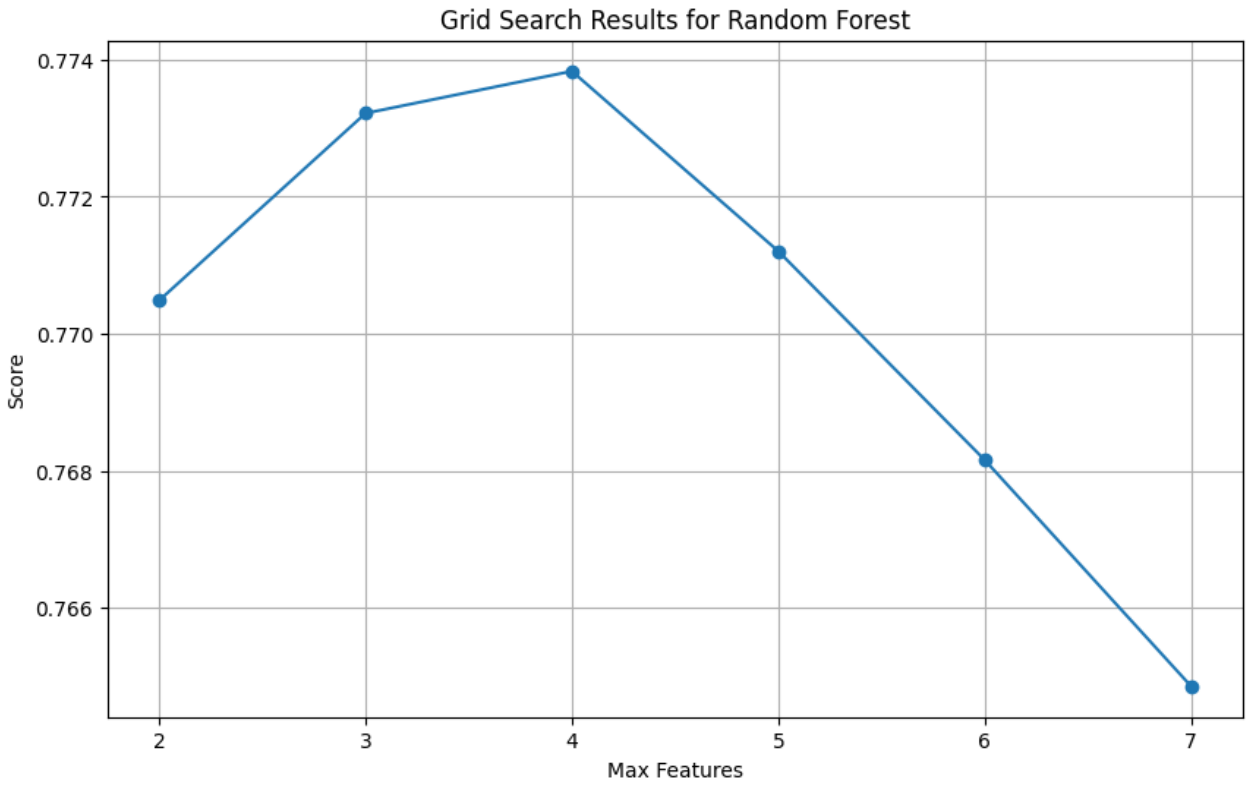
Figure 139. Random Forest - Confusion Matrix - Undersampling

The best performance is for “walk” class with a precision of 0.82, recall of 0.86 and an F1 score of 0.84. “Car” also performs well with a high precision of 0.93, 0.73 recall and an F1 score of 0.82. “Bike” has an average performance with 0.64 recall and 0.63 F1 score. The minority class, "Public transport," exhibits a high recall of 0.78, indicating that the model accurately identified 78.14% of the true labels

for that class. However, precision is low at 0.26, signifying that only 26% of the predictions made by the model as public transport truly belong to that class. Consequently, the F1 score is also modest at 0.41. The overall accuracy of the model stands at 0.74, while the F1 macro reaches 0.67. The biggest misclassification occurred between bike and car as the model erroneously predicted 15.58% of instances labelled as "car" to be "bike.".

For the second experiment, an oversampling strategy was employed. The pipeline, along with grid search for parameter tuning and resampling within cross-validation, was utilised once again. The initial parameter tuned was max_features, and the configuration is outlined below.

```
pipeline = Pipeline(steps = [
    ('scaler', MinMaxScaler()),
    ('sampler', RandomOverSampler(sampling_strategy='auto', random_state=42)),
    ('classifier', RandomForestClassifier(random_state=42, n_jobs=-1))
])
param_grid = {
    'classifier__max_features' : list(range(2, 8))
}
grid_search_rf = GridSearchCV(pipeline, param_grid, cv=skf, scoring='f1_macro')
```



The best score was found for **max_features = 4** and a validation **F1-macro of 0.78**. Following that, a subsequent grid search was carried out to investigate various combinations of both max_features and the number of estimators. The max_features parameter was kept at 4, identified as optimal in the prior search, while the number of trees varied within the range of [250, 350, 450]. The decision to explore a reduced number of parameter combinations was driven by the substantial increase of the dataset resulting from the oversampling strategy. This expansion introduced computational challenges in training the random forest, particularly when combined with the cross-validation process. The best score of the validation process was found for **max_features = 4** and **n_estimators = 450**. The oversampled model was then evaluated on the test set. The results are illustrated below.

Table 52. Random Forest Oversampling

Random Forest Oversampling	Precision	Recall	F1	Accuracy
Car	0.91	0.88	0.89	0.84
Bike	0.73	0.77	0.75	
Walk	0.86	0.89	0.87	
Public transport	0.76	0.53	0.63	
Macro Average	0.81	0.77	0.79	

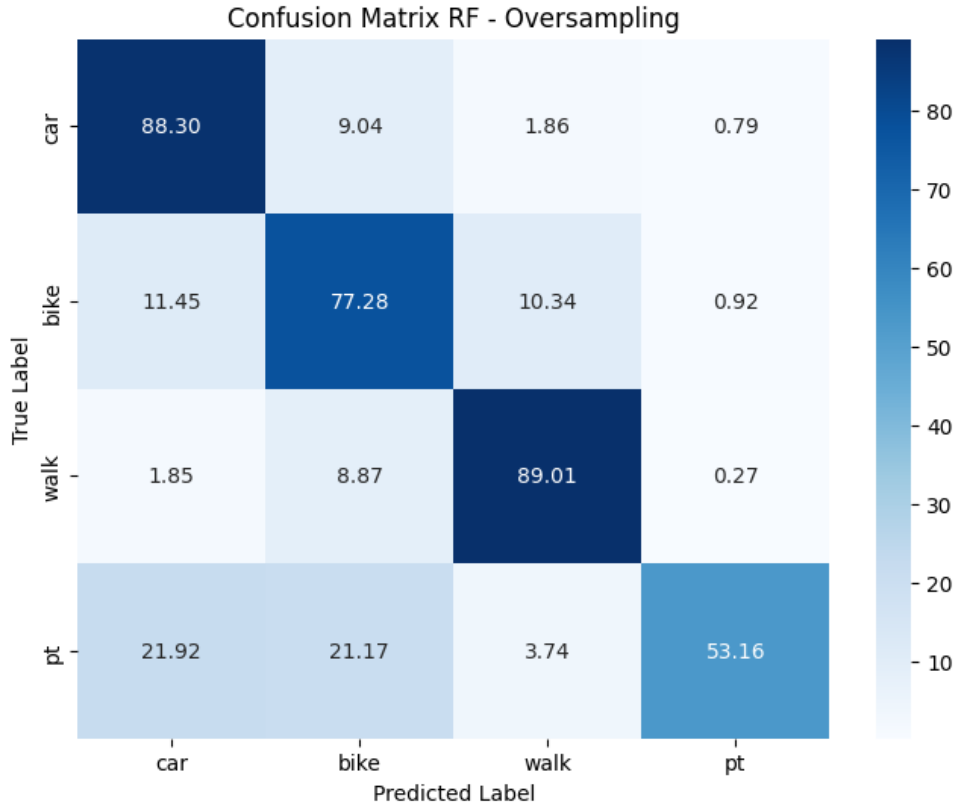


Figure 140. Random Forest - Confusion Matrix - Oversampling

The oversampled model exhibits an overall accuracy of 0.84, accompanied by an F1-macro of 0.79. Enhanced performances are observed for classes "car" and "walk," attaining F1 scores of 0.89 and 0.87, respectively. Notably, the metrics for class "bike" show a considerable improvement compared to the undersampled model, with precision, recall, and F1 scores reaching 0.73, 0.77 and 0.75 respectively. However, for the minority class, recall experiences a decrease from 0.78 to 0.53 compared to the undersampled model. In contrast, precision sees a significant increase from 0.28 to 0.76, and the F1 score rises from 0.41 to 0.63. Overall, the performance of random forest exhibits a substantial enhancement compared to the decision tree and logistic regression. Additionally, precision-recall curves were constructed as a supplementary evaluation metric for the models, as depicted below.

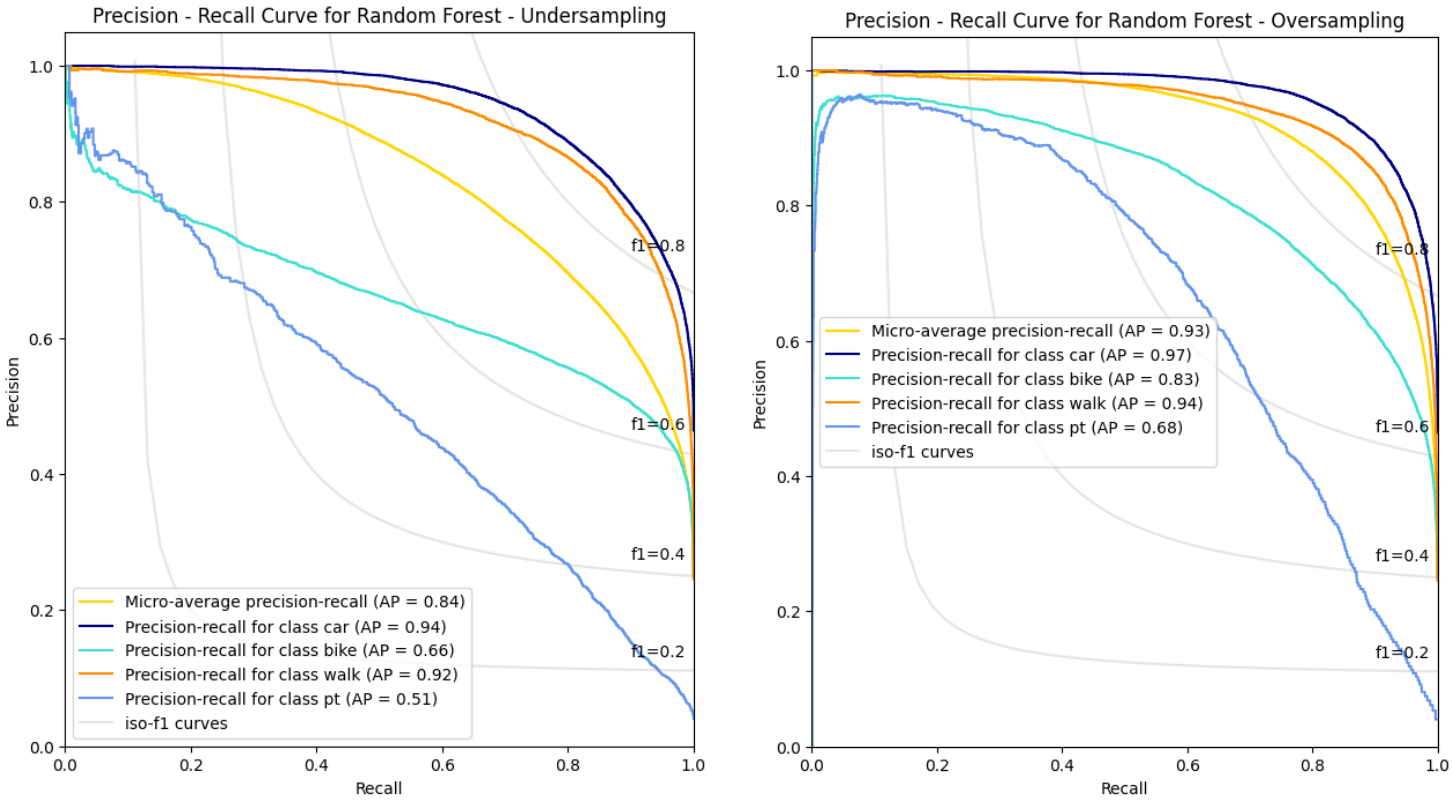


Figure 141. Precision Recall Curves - Random Forest

Upon examination of the curves, it is evident that the oversampling strategy has positively influenced the AP for each class. The most substantial improvement is observed for "bike," where AP has surged from 0.66 to 0.83. Similarly, for the minority class, AP has shown an increase from 0.51 to 0.68. Minor improvements also exist for car and walk classes in terms of AP. The models were also compared to the ones proposed by Kashifi et al. (2022).

Table 53. Model comparison - RF - Undersampling

Random Forest - Undersampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.88	0.54	0.56	0.31
Recall	0.63	0.63	0.70	0.84
F1 - score	0.74	0.58	0.63	0.45
AP	0.91	0.64	0.72	0.58
Micro-AP	0.73			
F1 - Macro	0.60			
Accuracy	0.654			

Random Forest - Undersampling **Thesis - Model**				
	Car	Bike	Walk	Public Transport
Precision	0.93	0.61	0.82	0.28
Recall	0.73	0.64	0.86	0.78
F1 - score	0.82	0.63	0.84	0.41
AP	0.94	0.66	0.92	0.51
Micro-AP	0.84			
F1 - Macro	0.67			
Accuracy	0.74			

Table 54. Model comparison - RF - Oversampling

Random Forest - Oversampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.90	0.83	0.83	0.84
Recall	0.94	0.84	0.77	0.57
F1 - score	0.92	0.84	0.80	0.68
AP	0.97	0.90	0.86	0.74
Micro-AP	0.93			
F1 - Macro	0.81			
Accuracy	0.87			

Random Forest - Oversampling **Thesis - Model**				
	Car	Bike	Walk	Public Transport
Precision	0.91	0.73	0.86	0.76
Recall	0.88	0.77	0.89	0.53
F1 - score	0.89	0.75	0.87	0.63
AP	0.97	0.83	0.94	0.68
Micro-AP	0.93			
F1 - Macro	0.89			
Accuracy	0.84			

When comparing the results to those presented in the paper, the oversampled model exhibits a notable increase only in the walk class, with all metrics showing a substantial improvement. Although the

metrics for other classes are consistently lower, they are not significantly behind. On the contrary, the undersampled model demonstrates an overall better performance, displaying increased metrics for all classes except the minority class (public transport). Finally, the feature importance was extracted for both undersampled and oversampled model. The results are illustrated below.

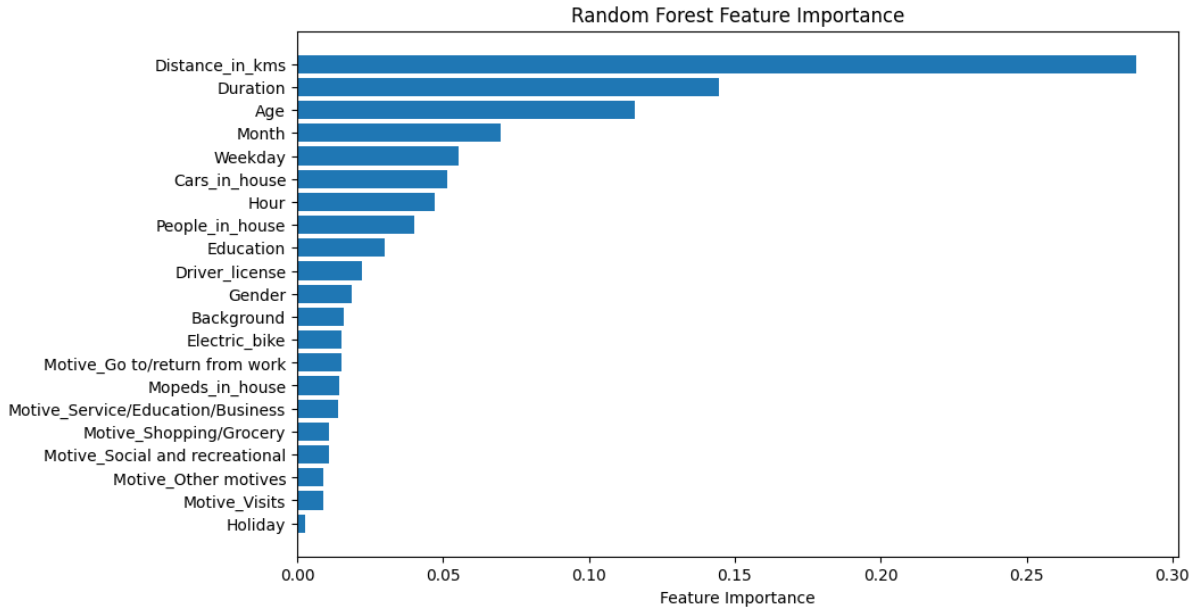


Figure 142. RF feature importance - Undersampling

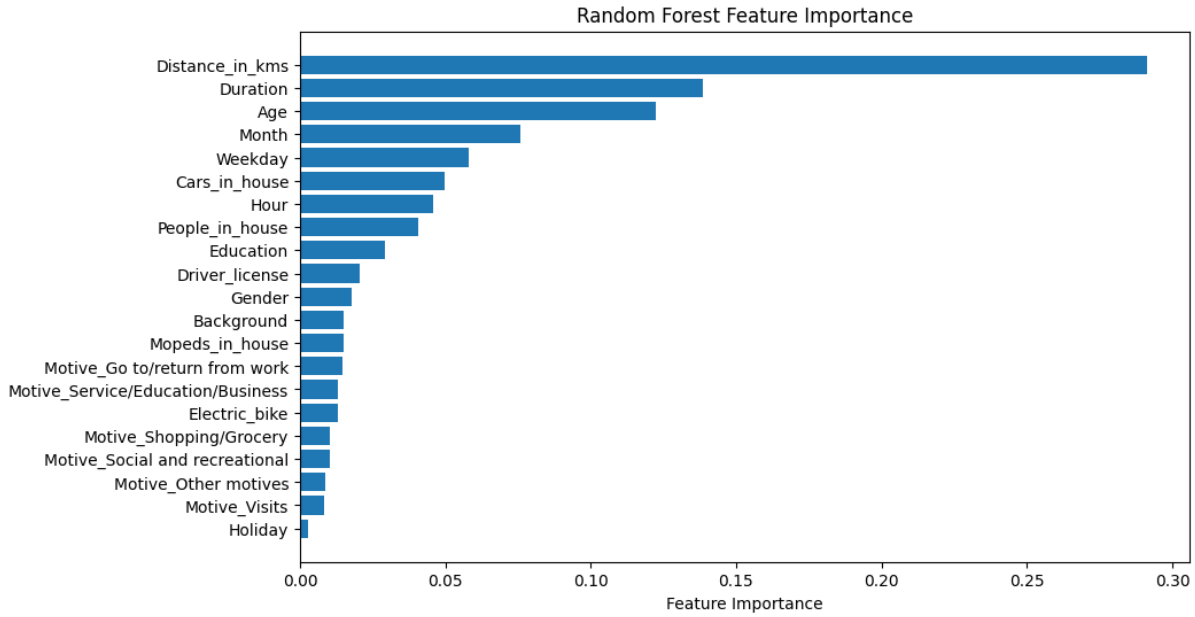


Figure 143. RF feature importance - Oversampling

In both models Distance emerges as the primary factor influencing the construction of individual trees, succeeded by Duration and Age. High on the list are also features such as Month, Weekday, and the number of cars. Conversely, whether the trip occurred on national holiday or not, ranks lowest on the list for both models.

4.2.8. XGBoost

The next model that was applied was XGBoost, experimenting with the undersampling strategy. The parameters that were tuned for the model were the learning_rate (the contribution of each tree), the subsample (the fraction of data for training) the colsample_bytree (fraction of features used for each tree) and the max depth parameter (tree size). For each of the parameters the value range was set within the interval [0.1 – 1], while for tree size the interval was [2, 16]. The pipeline was then created for resampling to take place within the cross-validation process of grid search. The configuration of the pipeline and the tuning results are illustrated below.

```
pipeline = Pipeline(steps = [
    ('scaler', MinMaxScaler()),
    ('sampler', RandomUnderSampler(sampling_strategy='auto', random_state=42)),
    ('classifier', xgb.XGBClassifier(objective='multi:softmax', num_class=4, random_state=42))
])
```

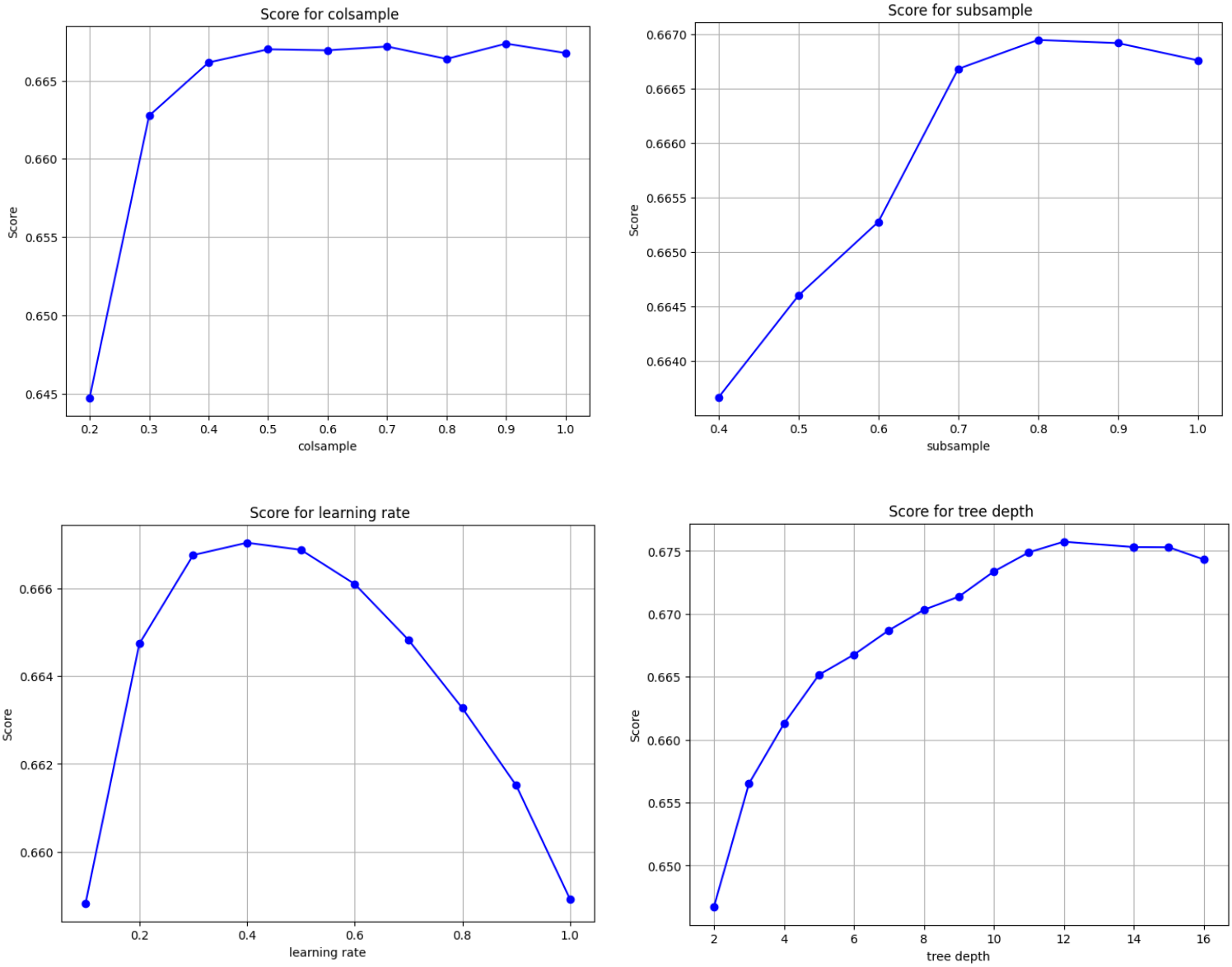


Figure 144. XGB Parameter tuning Undersampling

After reviewing the optimal areas, where the f1-macro score is higher, a final grid search was initiated while also increasing the number of estimators (trees). The parameter values are depicted below.

```
param_grid = {
    'classifier__learning_rate' : [0.3, 0.4],
    'classifier__subsample' : [0.7, 0.8],
    'classifier__n_estimators' : [250, 300, 350],
    'classifier__colsample_bytree' : [0.5, 0.7],
    'classifier__max_depth': [12]
}
grid_search = GridSearchCV(pipeline, param_grid, cv=skf, scoring="f1_macro", n_jobs=-1)
```

The optimal parameter values were **learning_rate = 0.3**, **colsample_bytree = 0.5**, **subsample = 0.8** and **n_estimators** = **300**, with a **validation f1 macro of 0.68**. The model was then evaluated on the test set. The results are illustrated on the classification report and the confusion matrix below.

Table 55. XGB Undersamping

XGBoost Undersampling	Precision	Recall	F1	Accuracy
Car	0.92	0.74	0.82	0.75
Bike	0.62	0.66	0.64	
Walk	0.82	0.84	0.83	
Public transport	0.30	0.80	0.44	
Macro Average	0.66	0.76	0.68	

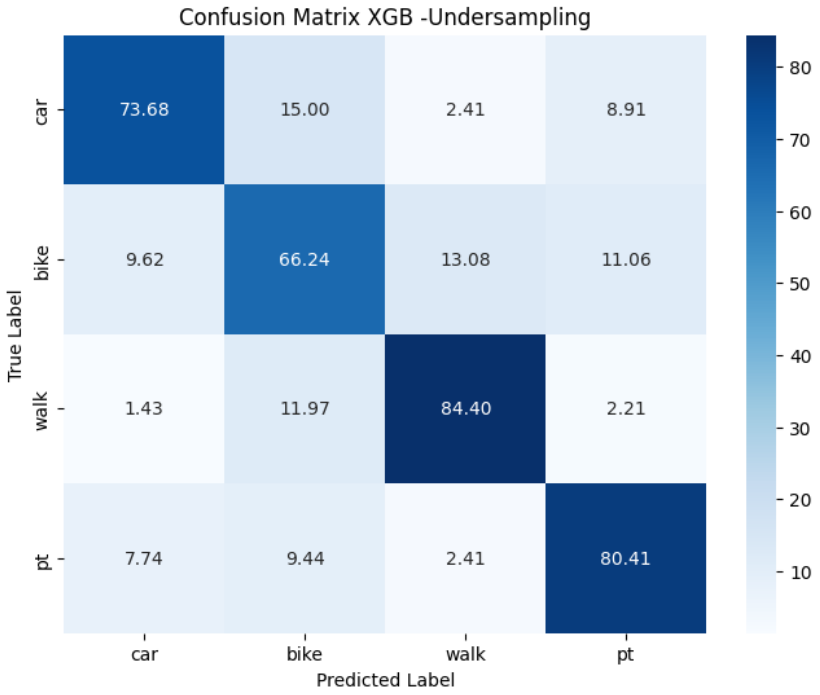


Figure 145. XGB - Confusion Matrix - Undersampling

The overall accuracy of the model is 0.75 while the f1-macro is 0.68. For “car”, the model demonstrates high precision (0.92), suggesting that when it predicts this class, it is accurate 93% of the time. However, the recall is relatively lower at 0.74, indicating that it misses 27% of actual car instances. “Bike” has a precision of 0.62, indicating moderate accuracy in positive predictions, while the recall is 0.66, suggesting it captures 66% of actual bike instances. Class “walk” exhibits high precision (0.82) and recall (0.84), resulting in a balanced F1-score of 0.83. In contrast, “public transport” has a precision of 0.30, indicating a high rate of false positives. However, it demonstrates high recall (0.80), suggesting it captures a significant portion of actual public transport instances. The F1-score is low at 0.44.

The second experimentation was with the oversampling strategy. The same parameters were tuned as the previous model. The pipeline configuration along with grid search results for the new oversampled model are depicted below.

```
pipeline = Pipeline(steps = [
    ('scaler', MinMaxScaler()),
    ('sampler', RandomUnderSampler(sampling_strategy='auto', random_state=42)),
    ('classifier', xgb.XGBClassifier(objective='multi:softmax', num_class=4, random_state=42))
])
```

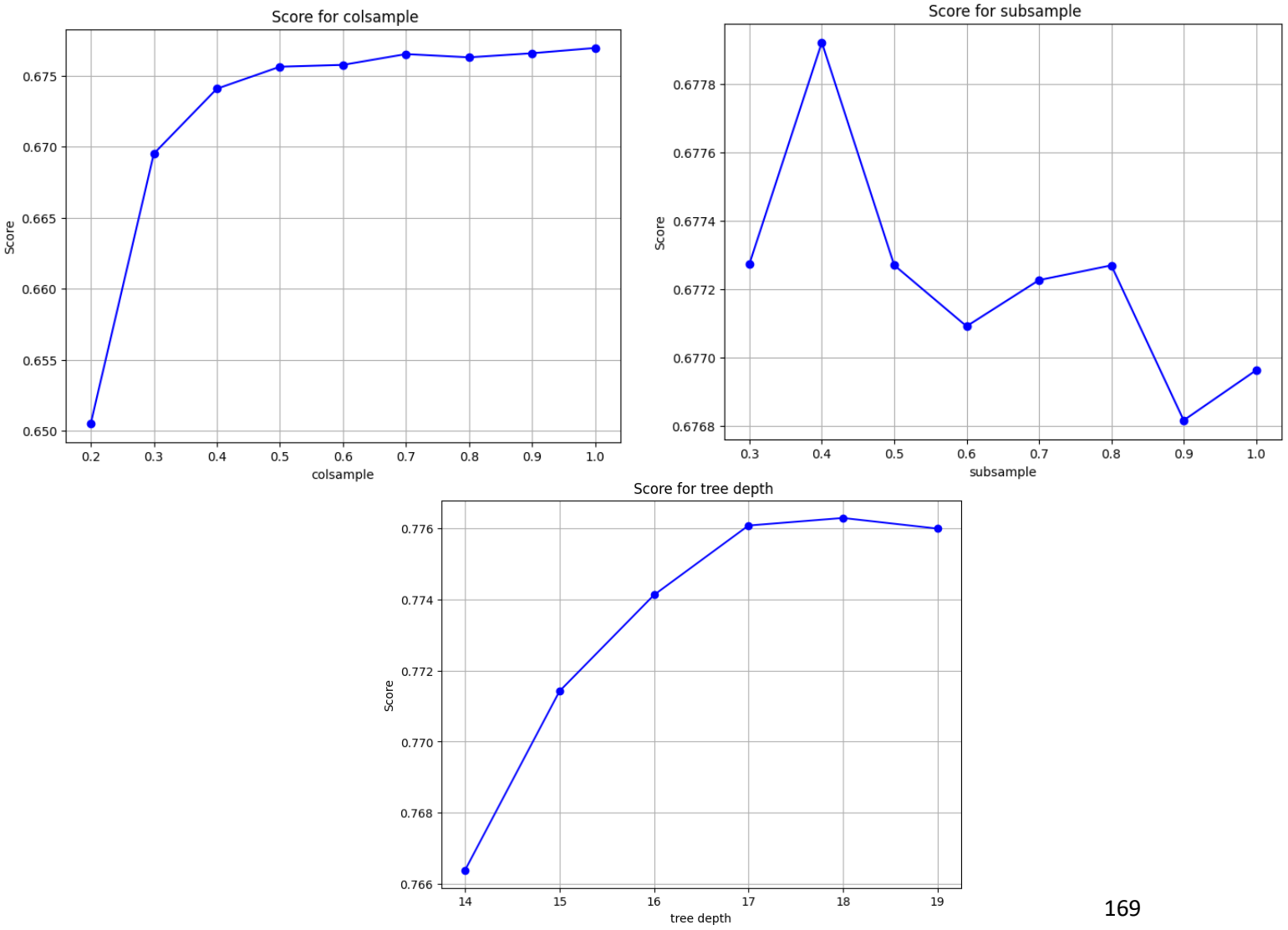


Figure 146. XGB Parameter tuning - Oversampling

After reviewing the optimal areas, where the f1-macro score is higher, a final grid search was initiated while also increasing the number of estimators (trees). The parameter values are depicted below.

```
param_grid = {
    'classifier__learning_rate' : [0.05, 0.1],
    'classifier__subsample' : [0.4],
    'classifier__n_estimators' : [600],
    'classifier__colsample_bytree' : [0.5],
    'classifier__max_depth': [18]
}
```

The best results were for **learning_rate = 0.1**, **colsample_bytree = 0.5**, **subsample = 0.4**, **n_estimators = 600** and **max_depth=18.** The model was then evaluated on the test set. The results are illustrated on the classification report and the confusion matrix below.

Table 56. XGB Oversampling

XGBoost Oversampling	Precision	Recall	F1	Accuracy
Car	0.92	0.89	0.90	0.86
Bike	0.75	0.80	0.77	
Walk	0.87	0.89	0.88	
Public transport	0.77	0.58	0.66	
Macro Average	0.83	0.79	0.80	

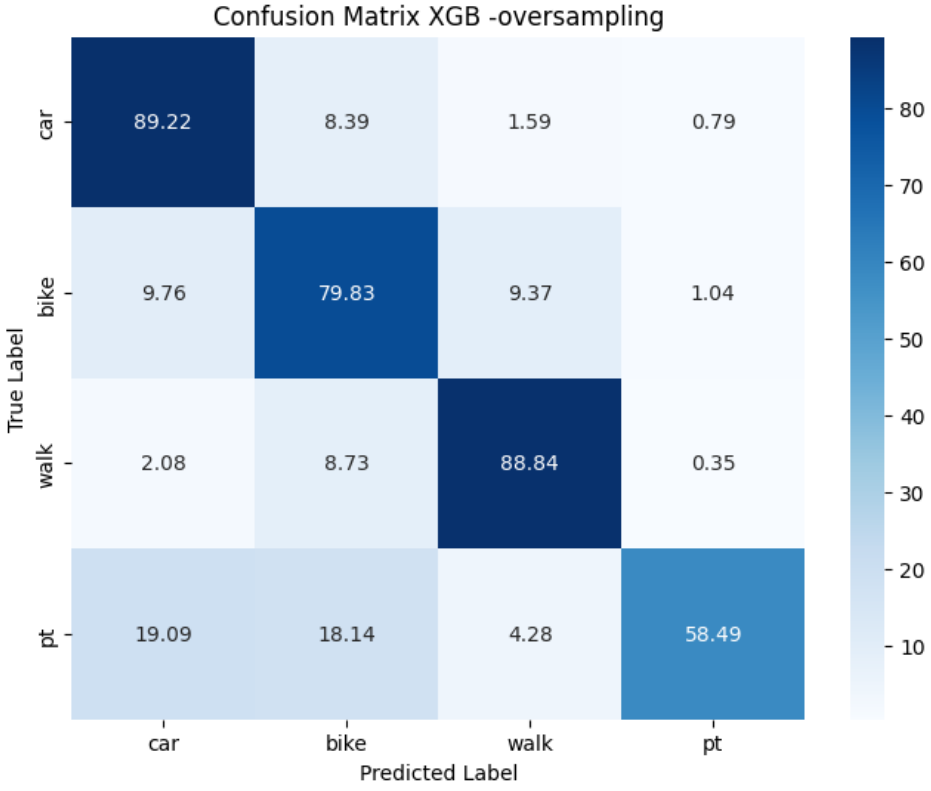


Figure 147. XGB - Confusion Matrix - Oversampling

The overall accuracy of the model is 0.86 while the F1-macro is 0.80. For “car”, the model excels with a high precision of 0.92, while a recall of 0.89 suggests capturing a substantial portion of actual instances, resulting in an overall balanced F1-score of 0.90. “Bike” demonstrates a decent balance between precision (0.75) and recall (0.80), yielding an F1-score of 0.77. “Walk” showcases strong performance with high precision (0.87) and recall (0.9), contributing to an impressive F1-score of 0.88. In contrast, “public transport” exhibits a trade-off, featuring a higher precision (0.77) with a moderate recall (0.58), leading to an F1-score of 0.66. To further assess the performance of the resampled models, precision-recall curves were constructed, as depicted below.

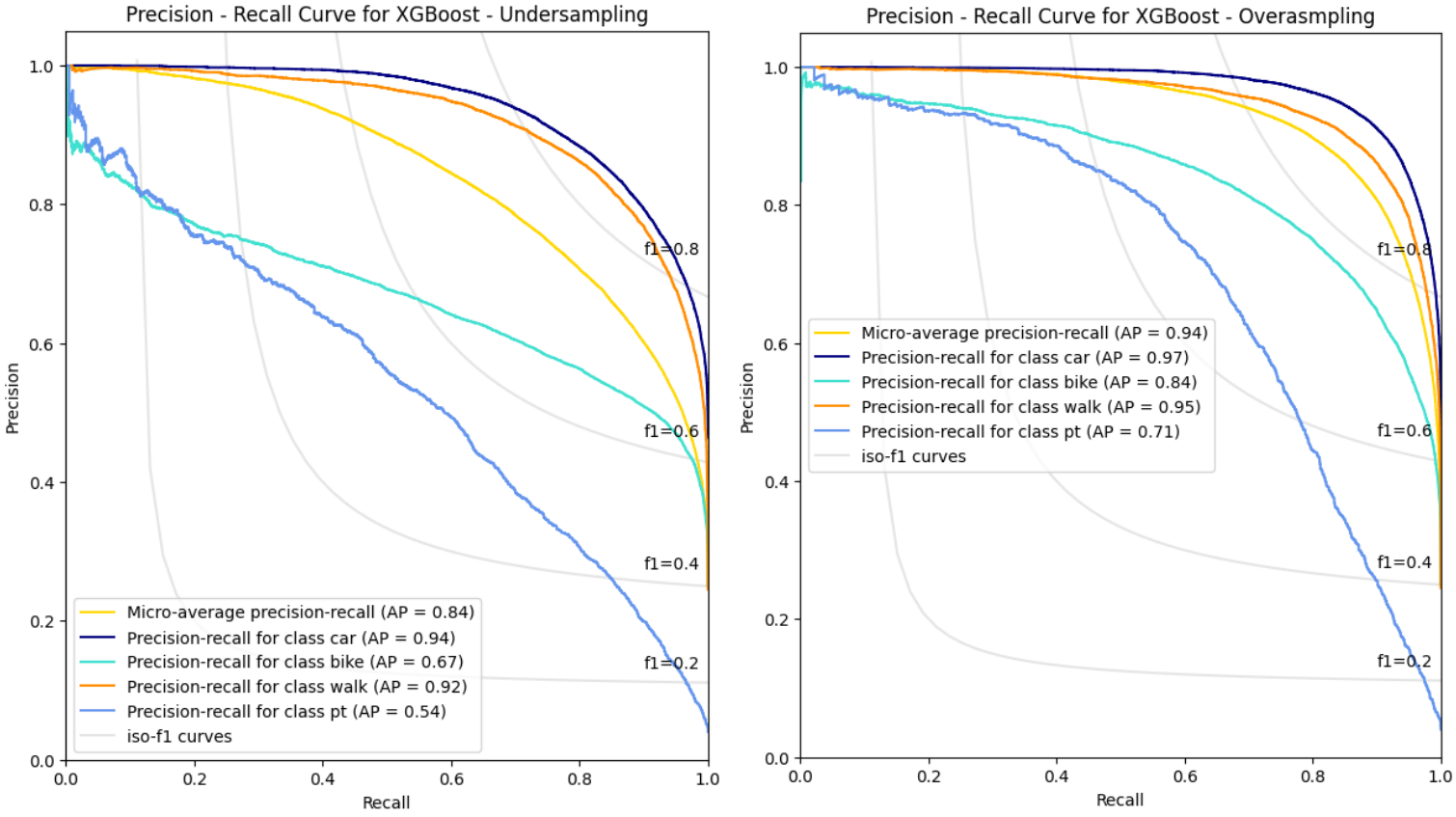


Figure 148. Precision Recall Curves - XGB

By observing the curves, it is evident that the oversampling strategy has positively influenced the area under the curve (AP) for each class. The most substantial improvement is observed for "bike," where AP has increased from 0.67 to 0.84. Similarly, for the minority class, AP has shown an increase from 0.54 to 0.71. Micro-AP also increased from 0.84 to 0.94. The models were also compared to the LightGBM models proposed by Kashifi et al. (2022).

Table 57. Model comparison XGB - Undersampling

LightGBM - Undersampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.90	0.55	0.56	0.34
Recall	0.66	0.66	0.70	0.88
F1 - score	0.76	0.60	0.63	0.49
AP	0.91	0.65	0.71	0.66
Micro-AP	0.76			
F1 - Macro	0.62			
Accuracy	0.675			

XGBoost - Undersampling				
	Car	Bike	Walk	Public Transport
Precision	0.92	0.62	0.82	0.30
Recall	0.74	0.66	0.84	0.80
F1 - score	0.82	0.64	0.83	0.44
AP	0.94	0.67	0.92	0.54
Micro-AP	0.84			
F1 - Macro	0.68			
Accuracy	0.75			

Table 58. Model comparison XGB - Oversampling

LightGBM - Oversampling **Model proposed by Kashifi et al. (2022)**				
	Car	Bike	Walk	Public Transport
Precision	0.92	0.85	0.85	0.87
Recall	0.95	0.87	0.87	0.68
F1 - score	0.93	0.86	0.86	0.76
AP	0.98	0.92	0.90	0.81
Micro-AP	0.95			
F1 - Macro	0.85			
Accuracy	0.89			

XGBoost - Oversampling				
	Car	Bike	Walk	Public Transport
Precision	0.92	0.75	0.87	0.77
Recall	0.89	0.80	0.89	0.58
F1 - score	0.90	0.77	0.88	0.66
AP	0.97	0.84	0.95	0.71
Micro-AP	0.94			
F1 – Macro	0.80			
Accuracy	0.86			

When comparing the results to those presented in the paper, the oversampled model exhibits a notable increase only in the walk class. In contrast, the undersampled model demonstrates an overall better

performance, displaying increased metrics for all classes except the minority class (public transport). The feature importances were also extracted for the two resampled models. The results are depicted below.

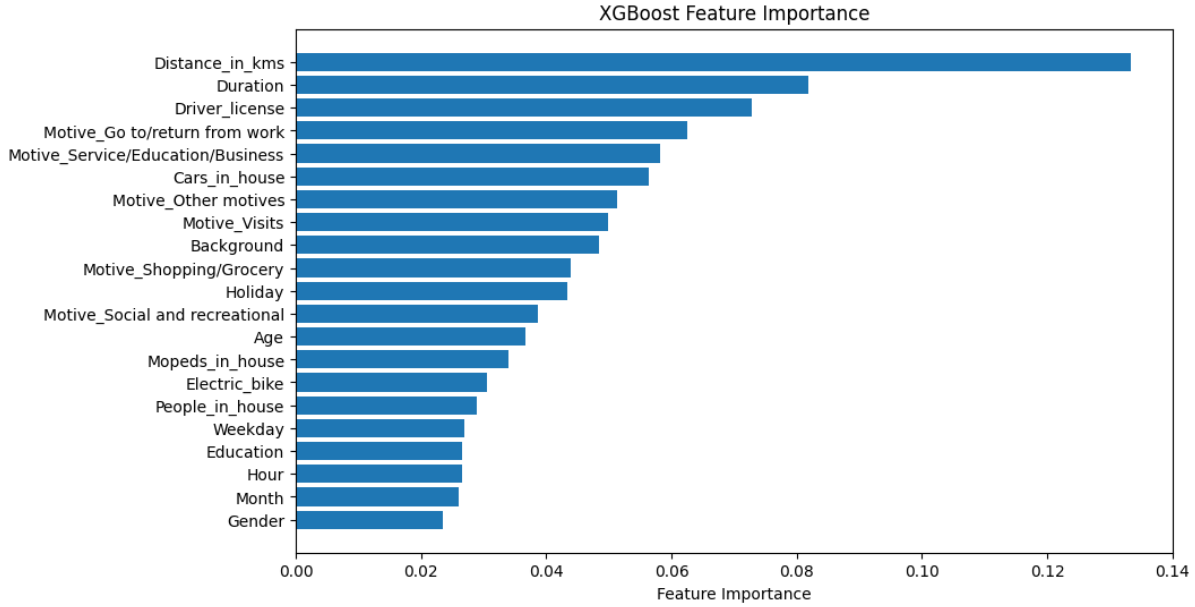


Figure 149. XGB feature importance - Undersampling

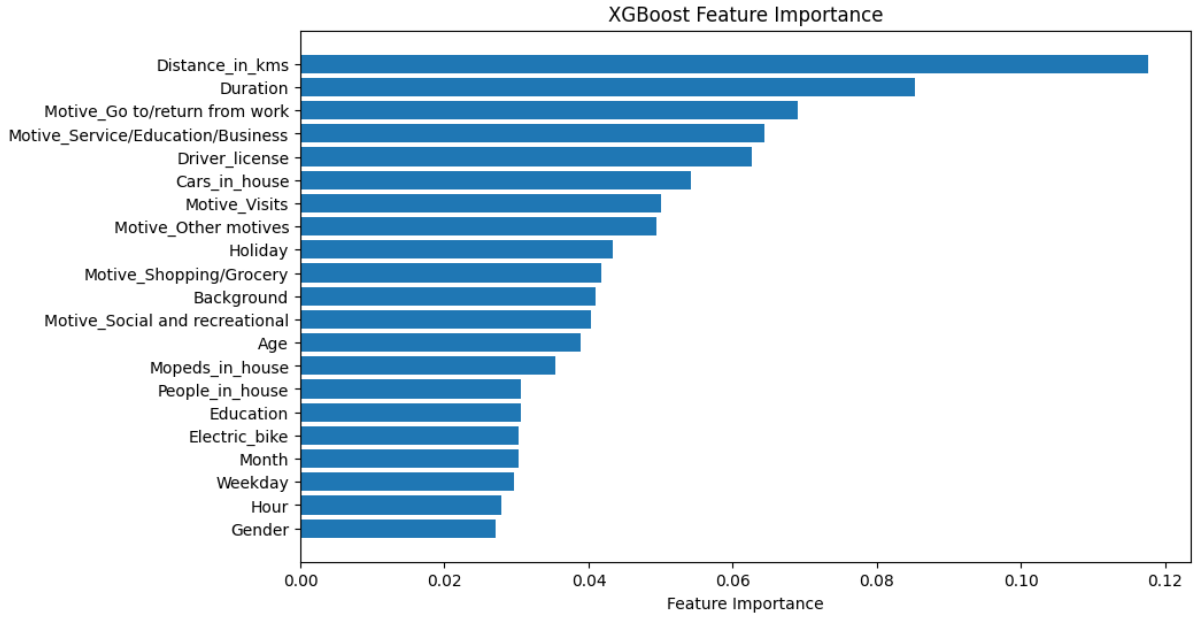


Figure 150. XGB feature importance - Oversampling

Distance and duration emerge as the most important feature for both models. For the undersampled model Driver licence is the third most important feature compared to Motive_go/return from home of the oversampled model. The least important features for both models are Gender, Hour, Month and Weekday. Comparing the results to the feature importance of Random Forest, XGBoost seems to be utilizing the binary features (Trip motives) a lot more for constructing the individual trees.

4.2.9. Model selection

To determine the best model, a comparative table was created, showcasing the average values of Precision, Recall, F-score across all classes, as well as micro-AP and accuracy for both undersampling and oversampling strategies.

Table 59. Model comparison undersampled and oversampled models

Undersampling Strategy				
	Logistic Regression	**Decision Tree**	**Random Forest**	**XGBoost**
Precision (Average)	0.58	0.62	0.66	0.66
Recall (Average)	0.64	0.68	0.75	0.76
F1-Macro	0.57	0.63	0.67	0.68
Micro-AP	0.72	0.77	0.84	0.84
Accuracy	0.65	0.71	0.74	0.75

Oversampling Strategy				
	Logistic Regression	**Decision Tree**	**Random Forest**	**XGBoost**
Precision (Average)	0.59	0.64	0.81	0.83
Recall (Average)	0.66	0.69	0.77	0.79
F1-Macro	0.59	0.65	0.79	0.80
Micro-AP	0.74	0.75	0.93	0.94
Accuracy	0.67	0.73	0.84	0.86

Undersampling strategy

- **Logistic Regression:** Logistic Regression exhibits a precision of 0.58, recall of 0.64, F1-Macro of 0.57, Micro-AP of 0.72, and an accuracy of 0.65. While it demonstrates reasonable recall, its precision and F1-Macro scores suggest room for improvement.
- **Decision Tree:** The Decision Tree model shows improvement over Logistic Regression with a precision of 0.62, recall of 0.68, F1-Macro of 0.63, Micro-AP of 0.77, and an accuracy of 0.71. It provides a more balanced trade-off between precision and recall.
- **Random Forest:** Random Forest exhibits notable improvement with a precision of 0.66, recall of 0.75, F1-Macro of 0.67, Micro-AP of 0.84, and an accuracy of 0.74. It outperforms both Logistic Regression and Decision Tree, particularly in terms of precision and overall accuracy.
- **XGBoost:** XGBoost emerges as the top performer among the considered models, highlighting a precision of 0.66, recall of 0.76, F1-Macro of 0.68, Micro-AP of 0.84, and the highest accuracy of 0.75. The model delivers a more balanced performance.

Oversampling strategy

- **Logistic Regression:** Logistic Regression, with oversampling, achieves a precision of 0.59, recall of 0.66, F1-Macro of 0.59, Micro-AP of 0.74, and an accuracy of 0.67. It shows improvement compared to the undersampling strategy, particularly in terms of precision and F1-Macro.
- **Decision Tree:** The Decision Tree model, with oversampling, demonstrates enhanced performance with a precision of 0.64, recall of 0.69, F1-Macro of 0.65, Micro-AP of 0.75, and an accuracy of 0.73. It maintains a balanced trade-off between precision and recall though micro-AP is reduced compared to the undersampled tree.
- **Random Forest:** Random Forest, under the oversampling strategy, exhibits improvement, illustrating a precision of 0.81, recall of 0.77, F1-Macro of 0.79, Micro-AP of 0.93, and an accuracy of 0.84. The model outperforms the individual tree and Logistic Regression model.
- **XGBoost:** XGBoost continues to be a top performer, even under the oversampling strategy, with a precision of 0.83, recall of 0.79, F1-Macro of 0.80, Micro-AP of 0.94, and the highest accuracy of 0.86. The model demonstrates the best overall performance.

Oversampling proves beneficial for all models, leading to improvements across various metrics compared to the undersampling strategy. Random Forest stands out with high precision, recall, F1-Macro, and accuracy. XGBoost remains the top performer, showcasing consistent performance across multiple metrics.

4.2.10. Encoding change and re-evaluation

In the previous chapters the models were applied on the dataset encompassing 21 features in total. However, Month, Weekday and Hour features were encoded into an ordinal scale (1, 2, 3, 4……etc). While the results were satisfactory, this encoding process could introduce bias to the models by assigning higher values to some categories (e.g. Monday = 1 vs Sunday = 7). To address that, those features were encoded using a different process. For Month, four new categories were created:

December, January, February → **Winter**

March, April, May → **Spring**

June, July, August → **Summer**

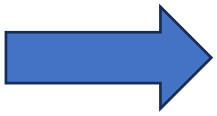
September, October, November → **Autumn**

Consequently, Month was encoded using a one hot encoding process created 4 new binary features for each season. The encoding for the "Weekday" feature involved representing instances occurring from Monday to Friday as 1, indicating trips on typical working days. In contrast, instances on Saturday and Sunday were encoded as 0, signifying trips outside typical working days (weekends). Therefore, "Weekday" has been converted into a binary feature with values of 1 and 0. Finally, “Hour” feature was also encoded using one hot encoding process creating 8 new binary features. The new feature list is illustrated below:

```
Index(['People_in_house', 'Gender', 'Age', 'Background', 'Education',
       'Driver_license', 'Cars_in_house', 'Mopeds_in_house', 'Electric_bike',
       'Weekday', 'Holiday', 'Round_trip', 'Duration', 'Distance_in_kms',
       'Motive_Go to/return from work', 'Motive_Other motives',
       'Motive_Service/Education/Business', 'Motive_Shopping/Grocery',
       'Motive_Social and recreational', 'Motive_Tours/hiking',
       'Motive_Visits', 'Month_Automn', 'Month_Spring', 'Month_Summer',
       'Month_Winter', 'Hour_01.00-04.00', 'Hour_04.00-07.00',
       'Hour_07.00-10.00', 'Hour_10.00-13.00', 'Hour_13.00-16.00',
       'Hour_16.00-19.00', 'Hour_19.00-22.00', 'Hour_22.00-01.00'],
      dtype='object')
```

The dimensionality has expanded to 33 features in total, a notable increase compared to the previous experiment's 21 features. To address multicollinearity, the Variance Inflation Factor (VIF) was recalculated. The initial iteration revealed some "inf" values, indicating multicollinearity issues. Consequently, the features round_trip, Month_winter, Motive_tours/hiking, and Hour_16.00-19.00 were removed. Afterward, the VIF values were computed again, and all of them were found to be below 5.

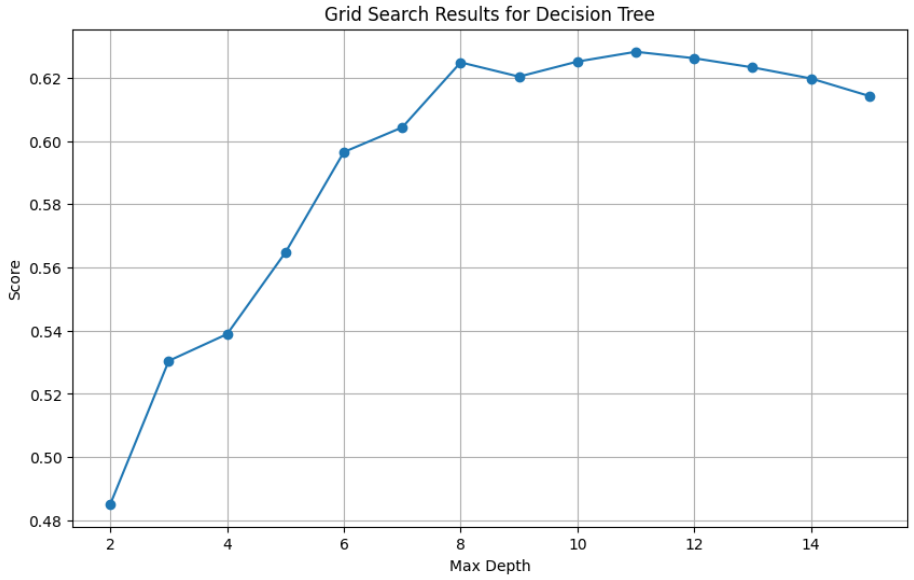
	Variable	VIF
0	const	0.000000
1	People_in_house	1.306296
2	Gender	1.026758
3	Age	1.268111
4	Background	1.093395
5	Education	1.040108
6	Driver_license	1.172860
7	Cars_in_house	1.301490
8	Mopeds_in_house	1.049185
9	Electric_bike	1.097913
10	Weekday	1.122558
11	Holiday	1.025221
12	Round_trip	5.827772
13	Duration	2.519663
14	Distance_in_kms	2.173967
15	Motive_Go to/return from work	inf
16	Motive_Other motives	inf
17	Motive_Service/Education/Business	inf
18	Motive_Shopping/Grocery	inf
19	Motive_Social and recreational	inf
20	Motive_Tours/hiking	inf
21	Motive_Visits	inf
22	Month_Automn	inf
23	Month_Spring	inf
24	Month_Summer	inf
25	Month_Winter	inf
26	Hour_01.00-04.00	inf
27	Hour_04.00-07.00	inf
28	Hour_07.00-10.00	inf
29	Hour_10.00-13.00	inf
30	Hour_13.00-16.00	inf
31	Hour_16.00-19.00	inf
32	Hour_19.00-22.00	inf
33	Hour_22.00-01.00	inf



	Variable	VIF
0	const	62.269107
1	People_in_house	1.306083
2	Gender	1.026695
3	Age	1.267753
4	Background	1.093390
5	Education	1.040093
6	Driver_license	1.172791
7	Cars_in_house	1.301257
8	Mopeds_in_house	1.049185
9	Electric_bike	1.097906
10	Weekday	1.122256
11	Holiday	1.025220
12	Duration	2.252735
13	Distance_in_kms	2.035543
14	Motive_Go to/return from work	3.469173
15	Motive_Other motives	2.509131
16	Motive_Service/Education/Business	2.280854
17	Motive_Shopping/Grocery	3.411739
18	Motive_Social and recreational	2.655282
19	Motive_Visits	2.255643
20	Month_Automn	1.541640
21	Month_Spring	1.513789
22	Month_Summer	1.509777
23	Hour_01.00-04.00	1.020457
24	Hour_04.00-07.00	1.121839
25	Hour_07.00-10.00	1.505861
26	Hour_10.00-13.00	1.547002
27	Hour_13.00-16.00	1.549553
28	Hour_19.00-22.00	1.346104
29	Hour_22.00-01.00	1.155701

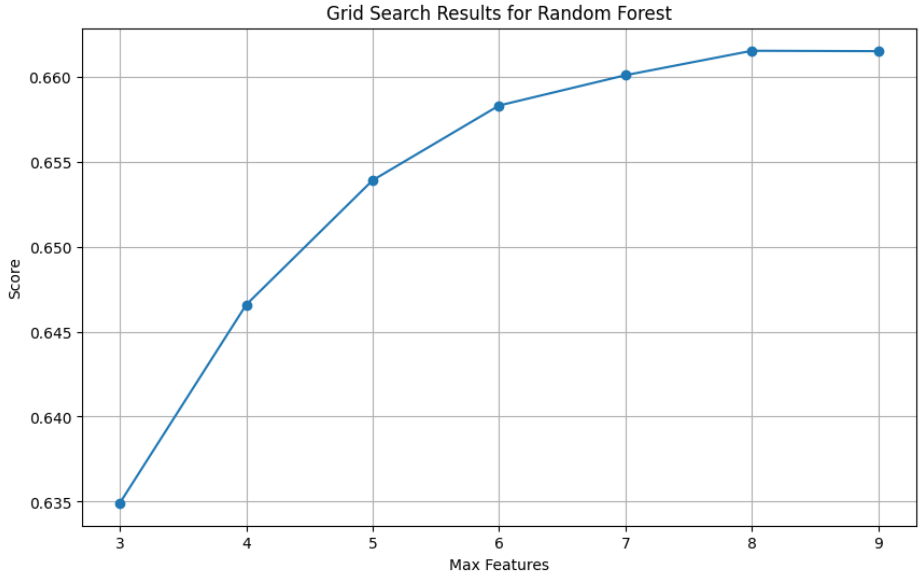
The same process was used as before. The pipelines were implemented to combine the scaler, the resampler (undersampling, oversampling) and the model in a chain sequence. The same four models were applied again: Logistic Regression, Decision Tree, Random Forest, XGBoost. First the undersampling strategy was implemented on the training set. The parameters tuned for the new models are illustrated below. The f1 macro was implemented as the scoring option for Grid Search while stratified kfold =10 was used for each fold to follow the same distribution.

Decision Tree



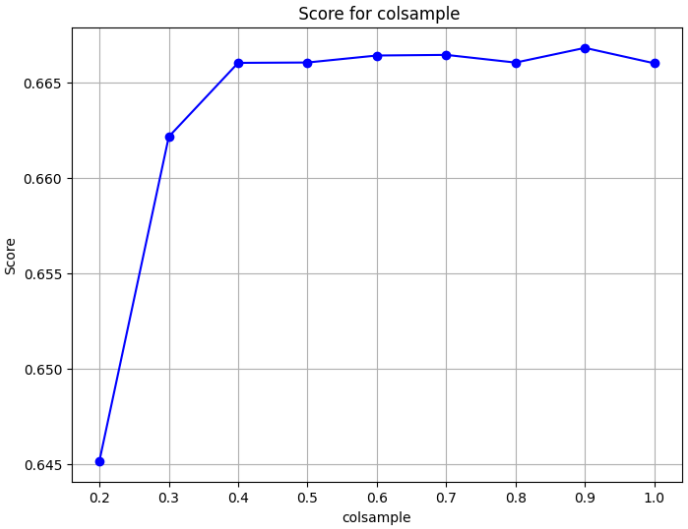
Random Forest

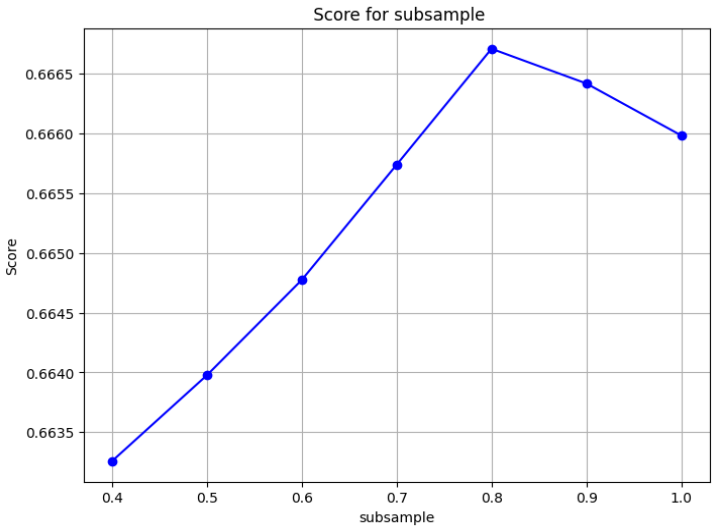
Number of estimators = 500

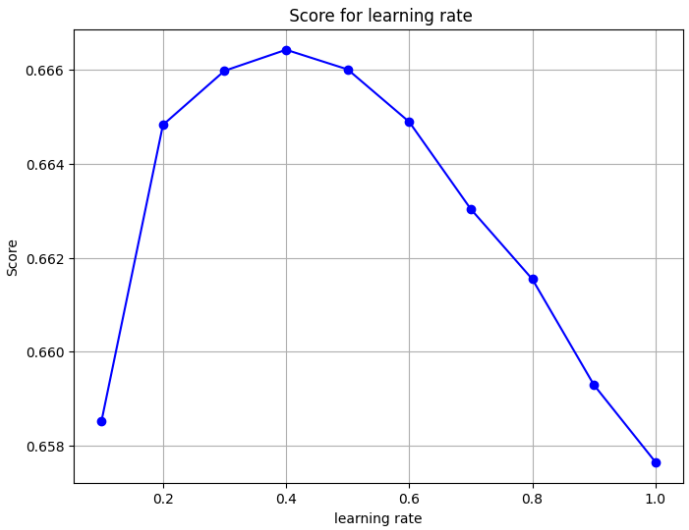


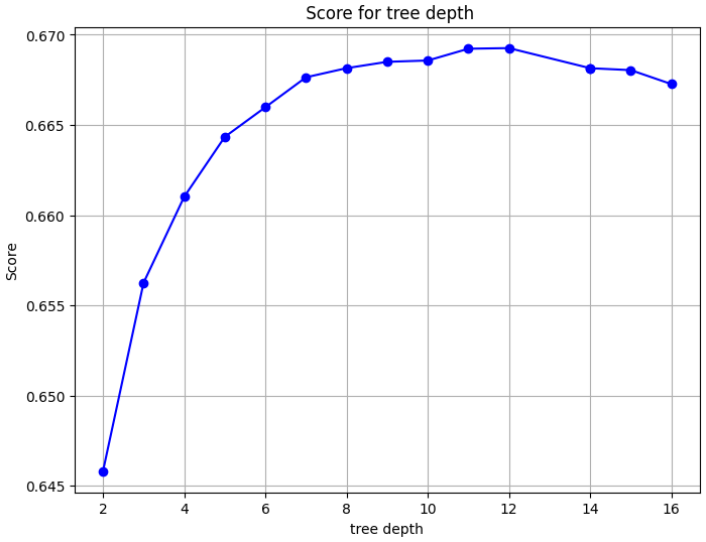
XGBoost

Number of estimators = 250







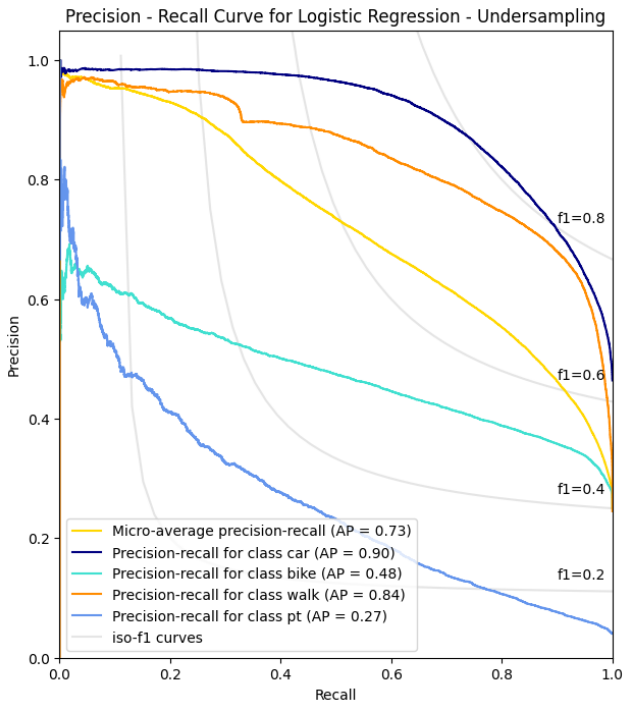


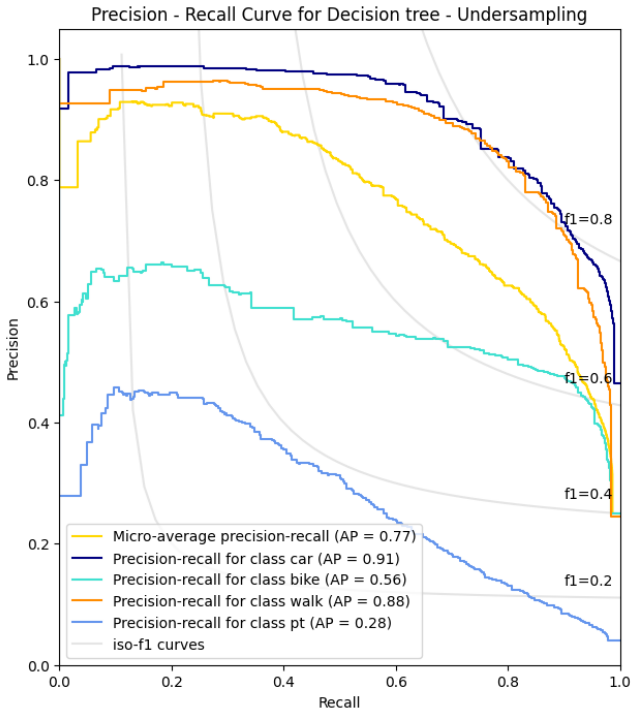
The undersampled models were then evaluated on the test set. The results are illustrated below.

Table 60. Model comparison (29 features) - Undersampling

Model	Mode	Precision	Recall	F1	Accuracy
Log.Reg	Car	0.88	0.70	0.78	0.65
	Bike	0.51	0.48	0.50	
	Walk	0.74	0.74	0.74	
	Public Transport	0.17	0.61	0.26	
Dec.Tree	Car	0.93	0.66	0.77	0.70
	Bike	0.53	0.66	0.59	
	Walk	0.82	0.82	0.82	
	Public Transport	0.22	0.64	0.33	
RF	Car	0.92	0.73	0.81	0.74
	Bike	0.60	0.64	0.62	
	Walk	0.82	0.86	0.84	
	Public Transport	0.27	0.76	0.40	
XGB	Car	0.92	0.73	0.81	0.74
	Bike	0.61	0.65	0.63	
	Walk	0.82	0.84	0.83	
	Public Transport	0.28	0.76	0.41	

XGBoost and Random Forest demonstrate the best performances overall with accuracy of 0.74. The metrics for the individual classes are also higher compared to Logistic regression and Decision tree. There is a trade-off for XGBoost and Random Forest, as the first demonstrates slightly better performance for the minority class, while the latest performs better for “walk” class. The precision recall curves were also constructed for the models as depicted below:





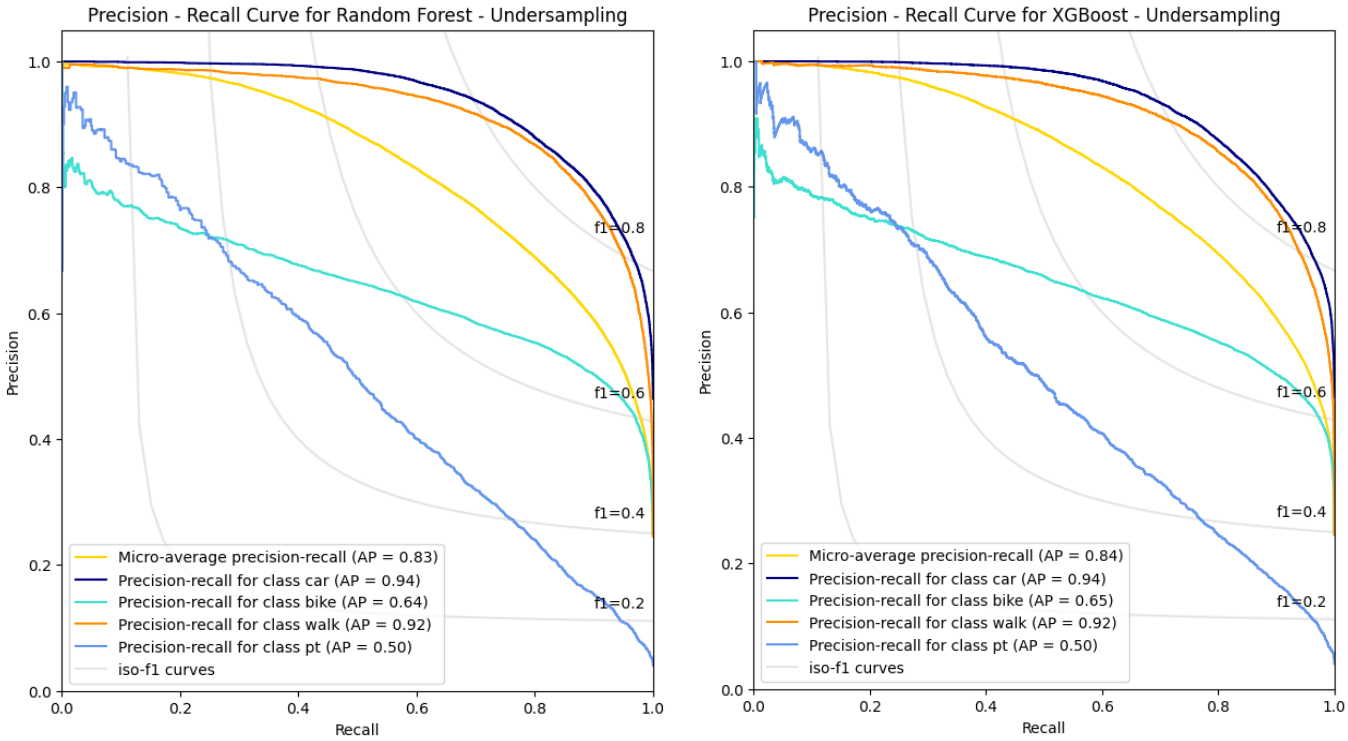
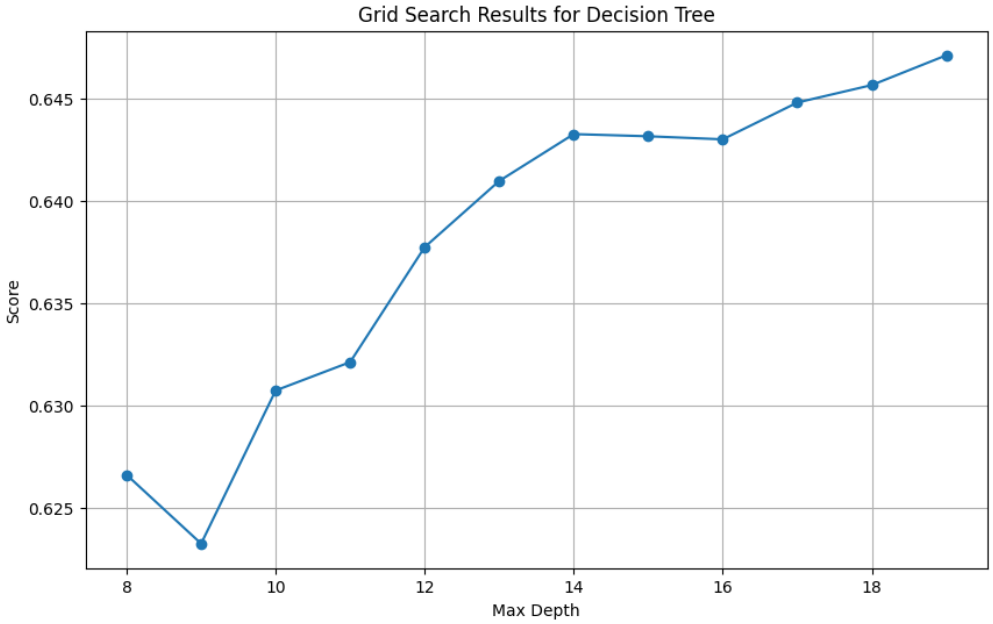


Figure 151. Precision Recall curves - Undersampled models

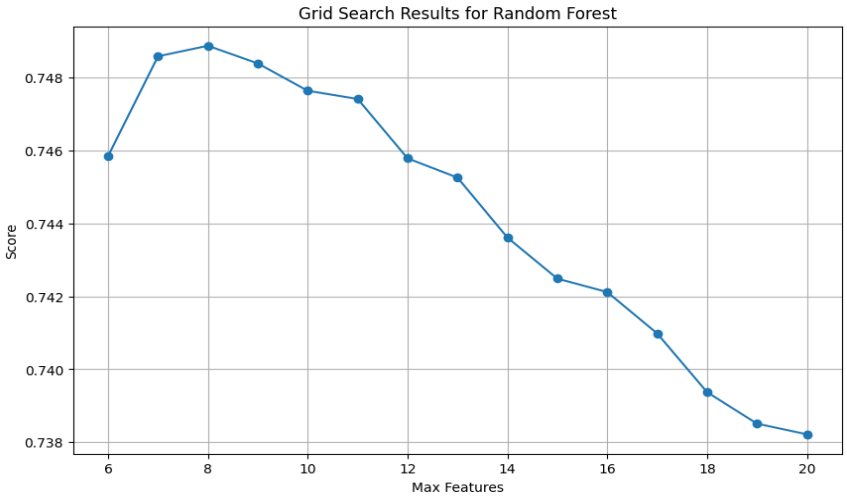
XGBoost demonstrates slightly increased AP for bike (0.65 vs 0.64), while the micro-AP is also higher compared to Random Forest (0.84 vs 0.83). The models were then evaluated experimenting with oversampling strategy. The parameter tuning for the new models is depicted below. The f1 macro was implemented as the scoring option for Grid Search while stratified kfold =10 was used for each fold to follow the same distribution.

Decision Tree



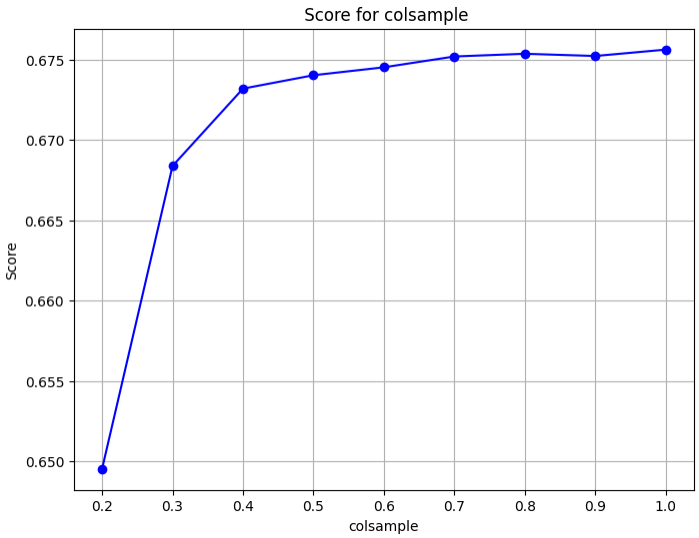
Random Forest

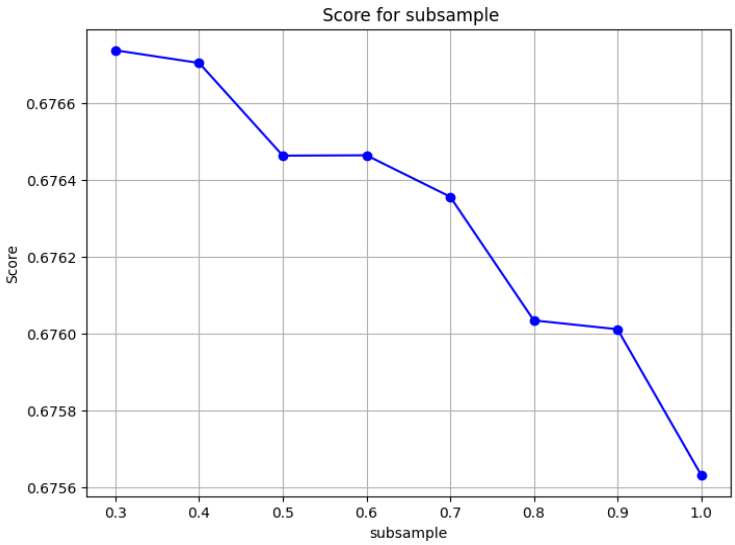
Number of estimators = 450

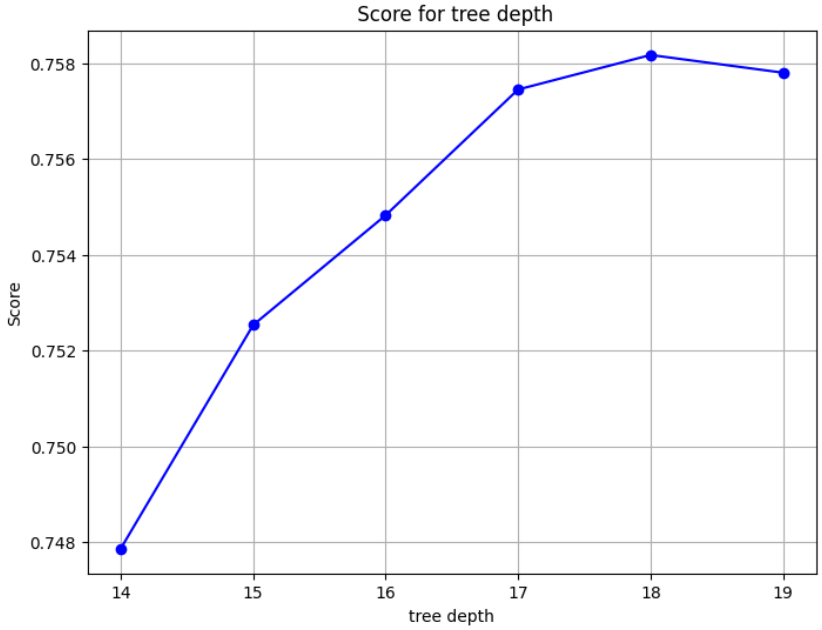


XGBoost

Number of estimators = 600, Learning_rate = 0.1







The oversampled models were then evaluated on the test set. The results are illustrated below.

Table 61. Model comparison (29 features) - Oversampling

Model	Mode	Precision	Recall	F1	Accuracy
Log.Reg	Car	0.88	0.71	0.79	0.67
	Bike	0.53	0.46	0.60	
	Walk	0.76	0.83	0.79	
	Public Transport	0.18	0.61	0.27	
Dec.Tree	Car	0.89	0.74	0.81	0.73
	Bike	0.58	0.68	0.62	
	Walk	0.82	0.81	0.82	
	Public Transport	0.27	0.52	0.35	
RF	Car	0.90	0.87	0.88	0.82
	Bike	0.69	0.74	0.72	
	Walk	0.85	0.88	0.86	
	Public Transport	0.73	0.50	0.59	
XGB	Car	0.90	0.88	0.89	0.83
	Bike	0.71	0.75	0.73	
	Walk	0.86	0.87	0.87	
	Public Transport	0.71	0.53	0.61	

XGBoost and Random Forest demonstrate the best performances overall with accuracy of 0.82 and 0.83 respectively. The metrics for the individual classes are also higher compared to Logistic regression and Decision tree. XGBoost performs slightly better than Random Forest when it comes to the minority class with Recall of 0.53 vs 0.50 and f1 score of 0.61 vs 0.59. The precision recall curves were also constructed for the models as depicted below:

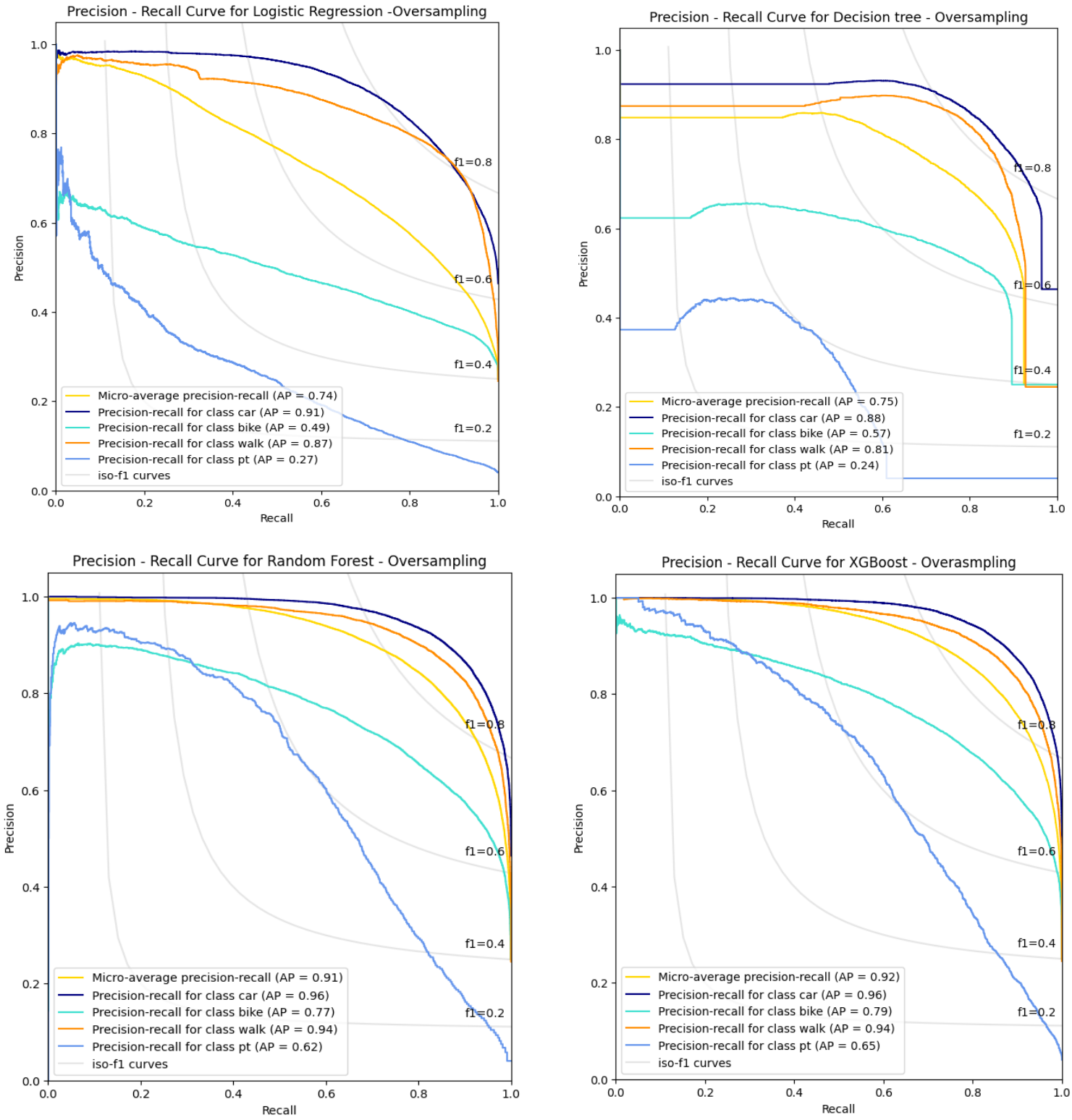


Figure 152. Precision Recall curves - Oversampled models

Random Forest and XGBoost models demonstrate increased performances compared to Logistic Regression and Decision Tree using an oversampling strategy. XGBoost showcases increased performance for bike and public transport classes, compared to Random Forest, as the AP are 0.79 vs 0.77 and 0.65 vs 0.62 respectively. The micro-AP is also higher for XGBoost with 0.92 compared to

0.91 of Random Forest. The table below illustrates the performance metrics on the test set for XGBoost using different encoding methods (29 vs 21 features) since it was found to be the best performing model.

Table 62. XGBoost model comparison

Undersampling		
	XGB (29 features)	**XGB (21 features)**
Precision (Average)	0.66	0.66
Recall (Average)	0.75	0.76
F1-Macro	0.67	0.68
Micro-AP	0.84	0.84
Accuracy	0.74	0.75

Oversampling		
	XGB (29 features)	**XGB (21 features)**
Precision (Average)	0.79	0.83
Recall (Average)	0.76	0.79
F1-Macro	0.77	0.80
Micro-AP	0.92	0.94
Accuracy	0.83	0.86

- **Undersampling:** For the XGBoost model employing 29 features, the precision and recall averages were 0.66 and 0.75, respectively. The F1-macro score is recorded at 0.67. The micro-AP has value of 0.84 with an overall accuracy of 0.74. In contrast, when the XGBoost model is applied with a reduced feature set of 21 features, the precision and recall averages demonstrate similar values of 0.66 and 0.75, respectively. The F1-macro score slightly improves to 0.68. The Micro-AP remains consistent at 0.84, reflecting a stable performance. Notably, accuracy is slightly increased at 0.75.

- **Oversampling:** For the XGBoost model employing 29 features under the oversampling strategy, the precision and recall averages were 0.79 and 0.76, respectively. Consequently, the F1-macro was 0.77. The micro-AP was found at 0.92 with an overall accuracy of 0.83. On the other hand, when the XGBoost model is applied with a reduced feature set of 21 features, the precision and recall averages demonstrate improved values of 0.83 and 0.79, respectively. The F1-macro score also increases to 0.80. The micro-AP further increases to 0.94 with an overall accuracy of 0.86.

5. CONCLUSIONS

5.1. Summary

In this chapter, the Thesis summarizes its findings, commencing with an overview of popular concepts, models, and evaluation metrics in machine learning derived from an extensive literature review. Notably, tree-based models emerge as the prevailing techniques for both classification and regression tasks, exhibiting heightened accuracy compared to traditional logit models, particularly in the realm of transport mode choice. The central focus of the Thesis lies in predicting the transport mode choice for two distinct cases. The first case focused and how residents in Thessaloniki commute to workplace. Data was collected through surveys, both online and in print, from active workers in the city, resulting in a total of 409 samples. After preparation and cleaning, 381 samples were deemed suitable for further analysis. The dataset comprises various trip-related features, such as distance, commute and departure time, alongside demographic characteristics like age, income, and household size. Ordinal scale features, like cost or physical activity impact on commutes, were also included on a scale of 1 to 5. Following an exploratory data analysis, preprocessing steps were implemented to make the data applicable for modelling in the Python programming language. These steps encompassed encoding, train/test splitting, and normalization of data within a 0-1 range.

Four models were employed in total: Decision Tree, Random Forest, XGBoost, and a Stacked model combining the previous three. Initial training involved 19 features, and various iterations were explored, including parameter tuning and distance metric swapping. The first experimentation, utilizing all features, revealed that Random Forest and XGBoost models exhibited the best performance, achieving overall accuracies of 0.92 and 0.93, respectively. However, substituting distance with the geodesic distance metric significantly diminished classifier performance. The most critical features for the models were identified as Distance, Duration, Health, Cost, Income, and Weather. As a secondary experiment, new models were constructed using only the most crucial features (7 in total). Excluding the Decision Tree, all models demonstrated improved performance, with Random Forest and XGBoost achieving overall accuracies of 0.93 and 0.94, respectively. Distance and time remained paramount among the features for these models. While the results are promising, the dataset is small, and no concrete decisions should be made. More data are essential for making the models robust.

Moreover, the SHAP explainer library was employed to further investigate feature importance for each class individually, providing a visualization of the model's decision-making process. Notably, XGBoost, being the best-performing model, identified distance and time as the primary influencers for the private vehicle class, while other classes exhibited minor distinctions. For the bus class, time emerged as the most crucial factor, whereas for the walk class, distance took precedence, followed by physical activity and health. The SHAP library proved to be a valuable tool in enhancing the understanding of how the model operates and what factors influence its predictions.

The second case was concentrated in predicting transport mode for residents in Netherlands. The data was collected from the Boreu of Statistics in Netherlands. The nature of the dataset was a trip diary encompassing the trips of residents throughout the country. The dataset was significantly larger than that of Thessaloniki, containing about 800000 trips and 20 features. After feature handling and data cleaning the resulting dataset consisted of **663745** samples. Trip related features were included in the dataset such as Distance, Duration, departure time and trip motive. Demographic features were also included about the respondents such as Age, gender, car ownership and household size. Preprocessing steps were implemented to make the data applicable for modelling in the Python programming language. These steps encompassed encoding, train/test splitting, and normalization of data within a 0-1 range. Since there was a class imbalance for the target variable undersampling and oversampling techniques were experimented. Though, to address this imbalance, python pipelines were constructed so to chain the scaling, resampling, model application along and parameter tuning all in a single process. Four models were applied in total: Logistic Regression, Decision Tree, Random Forest and XGBoost. All models were experimented with both resampling strategies. To access the performance of the models Precision, Recall, F1 score and AP (area under Precision – Recall curve) were considered since they are more suitable for imbalanced datasets. Additionally, the models were compared to the corresponding models proposed by Kashifi et al (2022), who also explored Netherlands transport behaviour using travel diary data from 2010-2012 and different set of features.

Employing an undersampling strategy, the optimal models were Random Forest and XGBoost, exhibiting superior performance across all metrics when compared to Logistic Regression and Decision Tree. XGBoost displayed slightly enhanced performance compared to Random Forest, with improvements in accuracy with 0.75, F1 macro of 0.68, and micro-AP of 0.84. This pattern persisted when implementing an oversampling strategy, where Random Forest and XGBoost consistently outperformed Logistic Regression and Decision Tree. Again, XGBoost showcased better results compared to Random Forest, featuring accuracy of 0.86, F1 macro of 0.80, and micro-AP of 0.94. Distance, duration and Age emerged as the most important features for Random Forest, while Distance, Duration, possession of driving license, number of cars in household and trip motives were highlighted for XGBoost regardless of resampling strategy.

5.2. Limitations and Future work

In this section of the Thesis, limitations and future research ideas are provided. Firstly, the case-specific nature of the models is a significant limitation, as their applicability may be confined to the specific contexts of Thessaloniki and the Netherlands. The ability to generalize these models to other cities or regions may be compromised due to variations in infrastructure, cultural factors, and transportation systems. Another limitation stems from class imbalance issues in the Netherlands datasets. The models may exhibit bias towards the majority class, potentially affecting their accuracy in predicting minority

classes. While key features such as distance, duration, and socio-demographic variables were identified as crucial for predicting transport mode, the generalization of these features to different contexts may not hold, as other regions may exhibit distinct factors influencing mode choices that were not considered in the studied cases. Additionally, the temporal limitations of the data used in this thesis may impact the models' predictive power over time, as urban mobility patterns evolve due to changes in infrastructure, technology, and societal preferences. Tailored to the unique commuting behaviour of Thessaloniki residents, the models are currently optimized for the classification of private vehicle, bus, and walk modes. To extend their applicability to additional modes such as bikes or skates, a broader and more diverse dataset would be essential. Furthermore, the dataset is relatively limited, and a more substantial amount of data is necessary to develop robust models for mode prediction.

To address these limitations and advance the research in predicting transport mode choices, several avenues for future work are proposed. Firstly, researchers could expand the scope of the models by validating them across multiple cities or regions with diverse characteristics. This would enhance the generalizability of the models and provide insights into differences in transport mode choices across different urban contexts. Integrating real-time and dynamic data sources, such as live traffic conditions, weather updates, and changes in city infrastructure, is suggested to enable the models to adapt to evolving urban mobility patterns and enhance prediction accuracy. Conducting detailed behavioural surveys to capture critical factors influencing transport mode choices, including psychological factors, perceptions of safety, and attitudes towards sustainable transportation, is recommended. Integrating such qualitative data can enrich the predictive capabilities of the models. Temporal analysis, considering features such as seasonal characteristics, weekly patterns, and daily variations, could provide a more comprehensive understanding of how commuting behaviours change over time. With the emergence of new transportation modes, researchers are encouraged to update the models to accommodate these evolving trends, ensuring their relevance in the face of ongoing changes in the urban transportation landscape. Lastly, investing in research to enhance the interpretability of machine learning models, particularly complex ensemble models like XGBoost, will foster greater trust in the models' predictions and facilitate a better understanding of feature importance. By addressing these areas of future work, researchers can contribute to the development of more robust and widely applicable models for predicting transport mode choices, ultimately fostering sustainable urban mobility solutions.

REFERENCES

Ababio-Donkor, A., Saleh, W. & Fonzone, A. 2020, "Understanding transport mode choice for commuting: the role of affect", *Transportation planning and technology*, vol. 43, no. 4, pp. 385-403.

Amiri, I.S., Akanbi, O.A. and Fazeldehkordi, E. (2014) *A Machine-Learning Approach to Phishing Detection and Defense*. 1st edn. Rockland, MA: Elsevier Science.

Amor, N.B., Benferhat, S. and Elouedi, Z. (2006) "Qualitative Classification with Possibilistic Decision Trees," in *Modern Information Processing*. Elsevier B.V, pp. 159–169.

Awan, A.A. (2023) *An introduction to shap values and machine learning interpretability*, *DataCamp*. Available at: https://www.datacamp.com/tutorial/introduction-to-shap-values-machine-learning-interpretability (Accessed: 27 February 2024).

Bei, H. et al. (2023) "Joint prediction of travel mode choice and purpose from travel surveys: A multitask deep learning approach," Travel, behaviour & society, 33, p. 100625.

Bellini, T. (2019) *IFRS 9 and CECL credit risk modelling and validation a practical guide with examples worked in R and SAS*. London: Academic Press.

Bhuiya, M.M.R. et al. (2022) "Application of Machine Learning Classifiers for Mode Choice Modeling for Movement-Challenged Persons," Future transportation, 2(2), pp. 328–346.

Böcker, L., Dijst, M. and Faber, J. (2016) "Weather, transport mode choices and emotional travel experiences," Transportation research. Part A, Policy and practice, 94, pp. 360–373.

Brownlee, J. (2020) *ROC curves and precision-recall curves for imbalanced classification*, *MachineLearningMastery.com*. Available at: https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-imbalanced-classification/ (Accessed: 20 February 2024).

Brownlee, J. (2021) Random oversampling and undersampling for imbalanced classification, MachineLearningMastery.com. Available at: https://machinelearningmastery.com/random-oversampling-and-undersampling-for-imbalanced-classification/ (Accessed: 12 January 2024).

Brownlee, J. (2021) *Stacking Ensemble Machine Learning with python*, *MachineLearningMastery.com*. Available at: https://machinelearningmastery.com/stacking-ensemble-machine-learning-with-python/ (Accessed: 20 February 2024).

Cunningham, P. and Delany, S.J. (2021) "k-Nearest Neighbour Classifiers - A Tutorial," *ACM computing surveys*, 54(6), pp. 1–25.

Das, S. *et al.* (2021) "Impact of COVID-19: A radical modal shift from public to private transport mode," *Transport policy*, 109, pp. 1–11.

Dash, S. (2022) *Decision trees explained-entropy, information gain, Gini Index, CCP pruning..*, *Medium*. Available at: https://towardsdatascience.com/decision-trees-explained-entropy-information-gain-gini-index-ccp-pruning-4d78070db36c (Accessed: 01 March 2024).

Hagenauer, J. and Helbich, M. (2017) “A comparative study of machine learning classifiers for modeling travel mode choice,” Expert systems with applications, 78, pp. 273–282.

Kettle, S. (2014) Geodesic distances: How long is that line again?, Esri Community. Available at: https://community.esri.com/t5/coordinate-reference-systems-blog/geodesic-distances-how-long-is-that-line-again/ba-p/902188 (Accessed: 25 January 2024).

Kubat, M. (2021) An Introduction to Machine Learning. 3rd edn. Cham: Springer International Publishing AG.

Kufel, J. et al. (2023) “What Is Machine Learning, Artificial Neural Networks and Deep Learning?-Examples of Practical Applications in Medicine,” Diagnostics (Basel), 13(15), p. 2582.

Lee, E. and Kim, S. (2023) *Geographic Information Systems for Intermodal Transportation*. Elsevier.

Lopez, F. (2021) Ensemble learning: Bagging & boosting, Medium. Available at: https://towardsdatascience.com/ensemble-learning-bagging-boosting-3098079e5422 (Accessed: 12 January 2024).

Lundberg, S. and Lee, S.-I. (2017) ‘A Unified Approach to Interpreting Model Predictions’, *arXiv.org* [Preprint]. Available at: https://doi.org/10.48550/arxiv.1705.07874.

Mayo, F.L. & Taboada, E.B. 2020, "Ranking factors affecting public transport mode choice of commuters in an urban city of a developing country using analytic hierarchy process: The case of Metro Cebu, Philippines", *Transportation research interdisciplinary perspectives*, vol. 4, pp. 100078.

McCarthy, L., Delbosc, A., Currie, G. & Molloy, A. 2017, "Factors influencing travel mode choice among families with young children (aged 0-4): a review of the literature", *Transport reviews*, vol. 37, no. 6, pp. 767-781.

McFadden, D. (1972). Conditional logit analysis of qualitative choice behavior.

McFadden, D. (1974) “The measurement of urban travel demand,” *Journal of public economics*, 3(4), pp. 303–328.

McFadden, D. and Train, K. (2000) “Mixed MNL models for discrete response,” *Journal of applied econometrics (Chichester, England)*, 15(5), pp. 447–470.

Misra, S., Li, H. & He, J. 2019, Machine Learning for Subsurface Characterization, 1st edn, Elsevier Science & Technology, San Diego.

Moped (2024) *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Moped (Accessed: 30 January 2024).

Mussone, L. & Changizi, F. 2023, "A study on the factors that influenced the choice of transport mode before, during, and after the first lockdown in Milan, Italy", *Cities*, vol. 136, pp. 104251-104251.

Narkhede, S. (2018) *Understanding AUC - roc curve*, *Medium*. Available at: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 (Accessed: 20 February 2024).

Omrani, H. (2015) ‘Predicting travel mode of individuals by machine learning’, *Transportation Research Procedia*, 10, pp. 840–849. doi:10.1016/j.trpro.2015.09.037.

Polamuri (2023) 7 most popular boosting algorithms to improve machine learning model's performance, Dataaspirant. Available at: https://dataaspirant.com/boosting-algorithms/ (Accessed: 12 January 2024).

Precision and recall (2024) *Wikipedia*. Available at: https://en.wikipedia.org/wiki/Precision_and_recall (Accessed: 04 March 2024).

Raschka, S. (2023) *Stackingcvclassifier: Stacking with cross-validation, StackingCVClassifier: Stacking with cross-validation - mlxtend*. Available at: https://rasbt.github.io/mlxtend/user_guide/classifier/StackingCVClassifier/ (Accessed: 20 February 2024).

Rocca, J. (2019) Ensemble methods: Bagging, boosting and stacking, Medium. Available at: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205 (Accessed: 11 January 2024).

Shanmugam, L. and Ramasamy, M. (2021) "Study on mode choice using nested logit models in travel towards Chennai metropolitan city," Journal of ambient intelligence and humanized computing, pp. Journal of ambient intelligence and humanized computing, 2021.

Souza, G.F.M.de *et al.* (2021) *Reliability Analysis and Asset Management of Engineering Systems*. 1st edn. San Diego: Elsevier (Advances in Reliability Science).

Statistics Netherlands (2024) *Dutch National Travel Survey*, *Statistics Netherlands*. Available at: https://www.cbs.nl/en-gb/onze-diensten/methods/surveys/korte-onderzoeksbeschrijvingen/dutch-national-travel-survey (Accessed: 29 January 2024).

Stefan Trueck, S.T.R. (2009) Rating Based Modeling of Credit Risk. 1st edn. San Diego: Elsevier Science.

svitla.org (2023) *Machine learning: Regression VS classification comprehensive guide, Svitla Systems - Global Digital Solutions Company*. Available at: https://svitla.com/blog/regression-vs-classification-in-machine-learning (Accessed: 01 March 2024).

Tamim Kashifi, M. et al. (2022) "Predicting the travel mode choice with interpretable machine learning techniques: A comparative study," Travel, behaviour & society, 29, pp. 279–296.

Trevisan, V. (2022) *Using shap values to explain how your Machine Learning Model Works*, *Medium*. Available at: https://towardsdatascience.com/using-shap-values-to-explain-how-your-machine-learning-model-works-732b3f40e137 (Accessed: 27 February 2024).

Wang, S., Mo, B. and Zhao, J. (2021) "Theory-based residual neural networks: A synergy of discrete choice models and deep neural networks," *Transportation research. Part B: methodological*, 146, pp. 333–358.

Willis, K.G. (2014) "The Use of Stated Preference Methods to Value Cultural Heritage," *Handbook of the Economics of Art and Culture*, 2, pp. 145–181.

XGBoost (2023) GeeksforGeeks. Available at: https://www.geeksforgeeks.org/xgboost/ (Accessed: 12 January 2024).

Yang, C., Fridgeirsson, E.A., Kors, J.A., Reps, J.M. & Rijnbeek, P.R. 2024, "Impact of random oversampling and random undersampling on the performance of prediction models developed using observational health data", Journal of big data, vol. 11, no. 1, pp. 1-17.

Zach (2021) *How to interpret Cramer's V (with examples)*, *Statology*. Available at: https://www.statology.org/interpret-cramers-v/ (Accessed: 11 March 2024).

Zhang, Z. et al. (2020) "A Customized Deep Neural Network Approach to Investigate Travel Mode Choice with Interpretable Utility Information," Journal of advanced transportation, 2020, pp. 1–11.

Zhao, X. et al. (2020) "Prediction and behavioral analysis of travel mode choice: A comparison of machine learning and logit models," Travel, behaviour & society, 20(C), pp. 22–35.

Zhou, ZH. (2021). Introduction. In: Machine Learning. Springer, Singapore. https://doi.org/10.1007/978-981-15-1967-3_1

APPENDIX 1

The full python code used for this Thesis is available at:

MariosMelach/Marios-Thesis: Transport Mode (github.com)

APPENDIX 2

This section presents the survey questions for Thessaloniki.

Μετακίνηση προς την εργασία στην πόλη της Θεσσαλονίκης

Αυτό το ερωτηματολόγιο αποτελεί ένα σημαντικό εργαλείο για τη συγκέντρωση πληροφοριών που θα συμβάλουν στην ενίσχυση της κατανόησής μας για την μετακίνηση προς στην εργασία στην περιοχή της Θεσσαλονίκης.

Τονίζουμε ότι η συμπλήρωση του ερωτηματολογίου είναι εντελώς εθελοντική, και κάθε απάντηση θα χρησιμοποιηθεί αποκλειστικά για σκοπούς έρευνας. Επίσης, επιθυμούμε να επισημάνουμε ότι έχουν ληφθεί όλα τα απαραίτητα μέτρα για τη διασφάλιση της ανωνυμίας και της εμπιστευτικότητας των συμμετεχόντων. Οποιεσδήποτε πληροφορίες που συλλέγονται δεν θα χρησιμοποιηθούν για οποιονδήποτε άλλο σκοπό εκτός από τον σκοπό της έρευνας και θα διατηρηθούν με απόλυτο σεβασμό προς την ιδιωτικότητα των συμμετεχόντων.

Σας ευχαριστούμε που θα διαθέσετε λίγο χρόνο για τη συμπλήρωση αυτού του ερωτηματολογίου και για τη συνεισφορά σας στην έρευνά μας.

* Υποδεικνύει απαιτούμενη ερώτηση

1. Διεύθυνση κατοικίας **(Οδός, αριθμός, ΤΚ)** *

2. Διεύθυνση εργασίας **(Οδός, αριθμός, ΤΚ ή πανεπιστήμιο για φοιτητές).** *

3. Φύλλο *

   Να επισημαίνεται μόνο μία έλλειψη.

   - ◯ Άνδρας
   - ◯ Γυναίκα

4. Ηλικία *

   Να επισημαίνεται μόνο μία έλλειψη.

   - ◯ 20 και κάτω
   - ◯ 21 - 30
   - ◯ 31 -40
   - ◯ 41 - 50
   - ◯ 51 - 60
   - ◯ 61 και άνω

5. Έχετε δίπλωμα οδήγησης για οποιοδήποτε από τα παρακατω? (**Επιλέξτε όλα όσα ισχύουν**) *

Επιλέξτε όλα όσα ισχύουν.

- [ ] Αυτοκίνητο
- [ ] Μηχανή
- [ ] Φορτηγό
- [ ] Δεν έχω δίπλωμα οδήγησης για κανένα όχημα

6. Διαθέτετε κάποιο από τα παρακάτω ? (**Επιλέξτε όλα όσα ισχύουν**) *

Επιλέξτε όλα όσα ισχύουν.

- [ ] Πατίνι
- [ ] Ποδήλατο
- [ ] Αμάξι
- [ ] Μηχανή
- [ ] Τίποτε από τα παραπάνω

7. Απο πόσα άτομα αποτελείται η οικογένειά σας ? (Να αναγράψετε **αριθμό**) *

_____

8. Πόσα ιδιωτικά οχήματα διαθέτετε στην οικογένειά σας? (Να αναγράψετε **αριθμό**) *

_____

9. Μηνιαίο Εισόδημα *

Να επισημαίνεται μόνο μία έλλειψη.

- ◯ 0 - 500
- ◯ 500 - 1000
- ◯ 1000 - 1500
- ◯ 1500 - 2000
- ◯ 2000 και άνω

10. Με ποιον τρόπο πηγαίνετε συνήθως στην εργασία σας (Ειτε σαν οδηγός είτε σαν επιβάτης). *

Να επισημαίνεται μόνο μία έλλειψη.

- ◯ Αμάξι
- ◯ Λεωφορείο
- ◯ Ποδήλατο
- ◯ Πατίνι
- ◯ Πόδια
- ◯ Μηχανή
- ◯ Άλλο: _____

11. Αναφέρετε πόσα λεπτά στο περίπου χρειάζεστε για να μεταβείτε από το σπίτι σας προς τον χώρο εργασίας σας. *

_____

12. Ποιές ώρες πηγαίνετε συνήθως στην εργασία σας? *

Να επισημαίνεται μόνο μία έλλειψη.

- ◯ 06.00 - 09.00
- ◯ 09.00 - 12.00
- ◯ 12.00 - 15.00
- ◯ 15.00 - 18.00
- ◯ 18.00 - 21.00
- ◯ 21.00 - 24.00
- ◯ 00.00 - 03.00
- ◯ 03.00 - 06.00

ΕΡΩΤΗΣΕΙΣ ΣΥΜΠΕΡΙΦΟΡΑΣ

Πιστεύετε πως οι παρακάτω παράγοντες επηρεάζουν τις μετακινήσεις σας?
Χρησιμοποιήστε κλίμακα όπου:
1) Διαφωνώ απόλυτα , 2) Διαφωνώ , 3) Ουδέτερο , 4) Συμφωνώ , 5) Συμφωνώ απόλυτα

13. Άνεση μετακίνησης *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
Διαι	◯	◯	◯	◯	◯	Συμφωνώ απόλυτα

14. Κόστος μετακίνησης *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
Διαι	◯	◯	◯	◯	◯	Συμφωνώ απόλυτα

15. Ασφάλεια μετακίνησης *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
Διαι	◯	◯	◯	◯	◯	Συμφωνώ απόλυτα

16. Προστασία του περιβάλλοντος *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
Διαφ	◯	◯	◯	◯	◯	Συμφωνώ απόλυτα

17. Σωματική άσκηση και υγεία *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
Διαφ	◯	◯	◯	◯	◯	Συμφωνώ απόλυτα

18. Καιρικές συνθήκες *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
Διαφ	◯	◯	◯	◯	◯	Συμφωνώ απόλυτα

19. Διαθεσημότητα χώρου στάθμευσης *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
Διαφ	◯	◯	◯	◯	◯	Συμφωνώ απόλυτα